

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

BOX, INC., and DROPBOX, INC.,
Petitioners

v.

TOPIA TECHNOLOGY, INC.,
Patent Owner

U.S. Patent No. 9,143,561
Filing Date: November 10, 2008
Issue Date: September 22, 2015

Title: Architecture for Management of Digital Files Across Distributed Network

U.S. Patent No. 10,067,942
Filing Date: September 21, 2015
Issue Date: September 4, 2018

Title: Architecture for Management of Digital Files Across Distributed Network

U.S. Patent No. 10,289,607

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Filing Date: June 25, 2018

Issue Date: May 14, 2019

Title: Architecture for Management of Digital Files Across Distributed Network

U.S. Patent No. 10,642,787

Filing Date: January 23, 2020

Issue Date: May 5, 2020

Title: Pre-File-Transfer Update Based on Prioritized Metadata

U.S. Patent No. 10,754,823

Filing Date: January 23, 2020

Issue Date: August 25, 2020

Title: Pre-File-Transfer Availability Indication Based on Prioritized Metadata

U.S. Patent No. 11,003,622

Filing Date: March 22, 2019

Issue Date: May 11, 2021

Title: Architecture for Management of Digital Files Across Distributed Network

DECLARATION OF RAVIN BALAKRISHNAN, PH.D.

Table of Contents

I.	INTRODUCTION AND QUALIFICATIONS	7
A.	Qualifications and Experience	7
B.	Materials Considered.....	13
II.	PERSON OF ORDINARY SKILL IN THE ART (“POSA”).....	15
III.	STATEMENT OF LEGAL PRINCIPLES.....	18
A.	Claim Construction.....	18
B.	Obviousness.....	20
IV.	OVERVIEW OF THE CHALLENGED PATENTS	25
A.	The Specifications of the Challenged Patents.....	26
B.	File Histories of the Challenged Patents	33
C.	The Challenged Claims of the Challenged Patents	38
V.	THE CHALLENGED CLAIMS OF THE ’823, ’787, ’607, ’622, ’942, AND ’561 PATENTS ALL RECITE SUBJECT MATTER THAT IS NOT DISCLOSED IN U.S. PROVISIONAL APPLICATION NO. 60/986,896	67
A.	The Challenged Claims of the ’823 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional	68
B.	The Challenged Claims of the ’787 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional	74
C.	The Challenged Claims of the ’607 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional	77
D.	The Challenged Claims of the ’622 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional	84
E.	The Challenged Claims of the ’942 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional	92

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

F.	The Challenged Claims of the '561 Patent Each Recite One or More Limitations that are Not Disclosed in the '896 Provisional	98
VI.	THE CHALLENGED CLAIMS OF THE '823, '787, '607, '622, '942, AND '561 PATENTS WOULD HAVE BEEN OBVIOUS IN VIEW OF SIGURDSSON IN COMBINATION WITH ONE OR MORE OF MOROMISATO, RAO AND/OR SHAPPELL	104
A.	Summary of the Prior Art.....	104
B.	'823 Ground 1: Claims 1-4, 8-11, and 13-16 of the '823 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Rao.....	123
C.	'823 Ground 2: Claims 1-4, 8-11, and 13-16 of the '823 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell, Rao, and Moromisato.....	198
D.	'787 Ground 1: Claims 1-4, 8-11, and 13-16 of the '787 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Rao.....	222
E.	'787 Ground 2: Claims 1-4, 8-11, and 13-16 of the '787 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell, Rao, and Moromisato.....	244
F.	'607 Ground 1: Claims 1, 3-5, 7, 12, 14, 17, 19, and 20 of the '607 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Rao.	250
G.	'607 Ground 2: Claims 1, 3-5, 7, 12, 14, 17, 19, and 20 of the '607 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell, Rao, and Moromisato.....	280
H.	'622 Sigurdsson Grounds: Claims 1, 3-5, 7, 11, 12, 14, 16, and 17 of the '622 Patent Would Have Been Obvious in View of Sigurdsson in Combination with One or More of Shappell, Rao, and Moromisato.....	292
I.	'942 Ground 1: Claims 1, 3, 4, 10, 12 and 13 of the '942 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell.....	299

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

J.	'942 Ground 2: Claims 1, 3, 4, 10, 12 and 13 of the '942 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Moromisato.	351
K.	'561 Ground 1: Claims 1, 3, 4, 8, 10 and 11 of the '561 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell.	361
L.	'561 Ground 2: Claims 1, 3, 4, 8, 10 and 11 of the '561 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Moromisato.	399
VII.	THE CHALLENGED CLAIMS OF THE '823, '787, '607, '622, '942, AND '561 PATENTS WOULD HAVE BEEN OBVIOUS IN VIEW OF BROWN IN COMBINATION WITH HESSELINK AND ONE OR MORE OF RAO AND/OR SHAPPELL	405
A.	Summary of the Prior Art.....	405
B.	'823 Ground 3: Claims 1-4, 8-11, and 13-16 of the '823 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink, Shappell, and Rao.....	416
C.	'787 Ground 3: Claims 1-4, 8-11, and 13-16 of the '787 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink, Shappell, and Rao.....	497
D.	'607 Ground 3: Claims 1, 3-5, 7, 12, 14, 17, 19, and 20 of the '607 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink, Shappell, and Rao.....	519
E.	'622 Brown Grounds: Claims 1, 3-5, 7, 11, 12, 14, 16, and 17 of the '622 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink, Shappell, and Rao...	548
F.	'942 Ground 3: Claims 1, 3, 4, 10, 12 and 13 of the '942 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink Alone (Claims 1, 3, 4, 10, 12 and 13) and in Further Combination with Shappell (Claims 4 and 13).	559
G.	'561 Ground 3: Claims 1, 3, 4, 8, 10 and 11 of the '561 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink.	601

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

VIII. NO SECONDARY CONSIDERATIONS OF NON-OBVIOUSNESS	633
IX. ENABLEMENT AND REASONABLE EXPECTATION OF SUCCESS	633
X. CONCLUSION.....	635

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

I, Ravin Balakrishnan, Ph.D., declare as follows:

I. INTRODUCTION AND QUALIFICATIONS

A. Qualifications and Experience

1. I am currently a tenured Full Professor of Computer Science in the Department of Computer Science at the University of Toronto. I joined the faculty at the University in 2001, was granted tenure in 2006, and served as Chair of my Department from 2015 to 2019. From 2006-2016, I held the Canada Research Chair in Human-Centered Interfaces in the Department of Computer Science.

2. I earned a B.Sc. in Computer Science from the University of New Brunswick in 1993, and an M.Sc. in Computer Science in 1997, and a Ph.D. in Computer Science in 2001, both from the University of Toronto. Since receiving my Ph.D., I have been a member of the faculty of the Department of Computer Science at the University of Toronto where my research has focused on human computer interaction (“HCI”), including the development of user interface technologies to improve HCI on a variety of computational platforms and application areas including mobile devices, web based technologies, data visualization and data analytics including web based analytics, touch input devices, video games, large displays, televisions, 3D displays, virtual and augmented reality technologies.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

3. In conjunction with my professorship, I serve as Co-Director of the Dynamic Graphics Project Laboratory at the University of Toronto that has twelve faculty members and roughly 50 graduate students and postdoctoral researchers working on various aspects of user interface interaction technologies, computer graphics, and virtual environments. Tech transfer of research to industry from this lab in recent years has resulted in several spin-off companies, including ones focused on 3D user interfaces, medical data visualization, virtual reality, and 3D architectural visualization and interaction.

4. As part of my professorial duties, I teach and supervise graduate students and postdoctoral fellows in their research work. To date, 27 research M.Sc. students, 15 Ph.D. students, and 9 postdoctoral fellows have completed their research training under my guidance. Their research has led to theses and peer reviewed publications over a broad range of topics covering interactive computing and user interfaces for multiple applications and technology platforms. I have taught courses covering computing topics including interactive computing, human-computer interaction, and information systems and design, as well as courses covering various user interface technologies for a variety of platforms including web-based systems.

5. I have published over 140 refereed publications in peer-reviewed

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

journals and conferences. I have further presented numerous conference abstracts, posters, talks, and demonstrations in my field. My work has been cited more than 20,000 times across a spectrum of high-impact publications, and my current h-index score is 81, which is an indication of the significant impact of my research on scholarly literature in the field of Computer Science.

6. I have received major awards and honors in my field. For example, in 2007, I received an Alfred P. Sloan Research Fellowship. In 2011, I was elected to the Association for Computing Machinery's Computer Human Interaction Academy, which is an honorary group consisting of researchers who have made extensive contributions to the study of HCI and who have led the shaping of the HCI field. The stated criteria for being elected to the Association for Computing Machinery's Computer Human Interaction Academy are: (1) cumulative contributions in the field; (2) impact on the field through development of new research directions and/or innovations; and (3) influence on the work of others. In 2020, I was presented with the Canadian Human Computer Communications Society's Achievement Award, which recognizes researchers who have made a substantial contribution to the fields of computer graphics, visualization, or human-computer interaction.

7. I have been a co-founder on a number of start-ups focusing on human-

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

computer interfaces, including Bump Technologies (acquired by Google in 2010), which developed a physically realistic desktop user interface system, FindSpace (formerly Sketch2 Corp. and Arcestra), which developed software for designing 3D interior layouts (acquired by Cougar Group in 2021), and Conceptualiz, which develops surgical planning technology using primarily a touch interface.

8. My research at the University of Toronto has involved nearly every broad aspect of human-computer interaction and data visualization. For instance, I have done significant work in the areas of input devices, displays, sensing technologies, interfaces to small and/or mobile computers, interfaces to displays of the future, data analytics including web analytics, and interaction techniques, including touch and multi-touch, gestural, sketching, and multi degree-of-freedom interactions. As another example, I have done work in the evaluation of user interfaces, including associated metrics and predictive models of human performance.

9. In addition to my research at the University of Toronto, I have collaborated with researchers at leading institutions worldwide. For example, I have been a visiting professor at Laboratoire de Recherche en Informatique (LRI) at the Université Paris-Sud, France. Additionally, I have been a visiting researcher at several industrial laboratories, including: (1) HP Labs, (2) Mitsubishi Electric

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Research Laboratories (MERL), and (3) Microsoft Research Labs. At those companies, my work generally focused on developing new user interface techniques for a variety of technology platforms, including work on new data analytics and visualization systems. Prior to becoming a professor, during my MSc and PhD studies, I was also concurrently a part-time researcher at Alias | Wavefront (now part of Autodesk), that was a leading developer of 3D graphics software applications. Some of my work at these companies, and my startup companies, have resulted in software and hardware implementations that were shipped within consumer level products.

10. I have also served on the organizing and paper reviewing committees of many leading conferences in my field and have taken on editorial roles for leading technical journals in fields pertinent to my research. For example, I have been an associate editor of the Association for Computing Machinery (“ACM”) Transactions on Computer-Human Interaction and an associate editor of the IEEE Transactions on Visualization and Computer Graphics, which are both peer-reviewed journals. Similarly, I have been the paper’s chair for the ACM UIST Symposium on User Interface Software and Technology and have served multiple times as an associate chair for the ACM CHI Conference on Human-Computer Interaction. Over my career, I have also reviewed hundreds of published and unpublished papers in my

field.

11. I am the co-inventor on 20 patents, and have several patent applications pending, all in the area of user interfaces, including new techniques and technologies for handling user input to computational platforms in novel and more facile ways.

12. Additionally, I have served as an expert witness in numerous patent actions, including in multiple IPR proceedings and ITC and district court cases, including for Apple in several actions against Samsung, HTC and other entities involving multiple patents covering tablet and mobile phone interfaces, for Nintendo involving their Wii gaming technology, for Immersion involving their haptic feedback technology, for Facebook/Oculus involving their virtual reality technology, and for Rovi/Tivo involving their various TV guide, video on-demand, and set-top box technologies. A more detailed list of my qualifications and a listing of the cases in which I have served as an expert witness, including where I have testified at trial or by deposition, can be found in my Curriculum Vitae, which is attached as Ex. A.

13. I am being compensated for my time working on this matter at my standard hourly rate plus expenses. I do not have any personal or financial stake or interest in the outcome of the present proceeding, and the compensation is not dependent on the outcome of this IPR and in no way affects the substance of my

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

statements in this Declaration.

B. Materials Considered

14. The analysis that I provide in this Declaration is based on my education, research, and experience, as well as the documents I have considered. In forming my opinions, I have read and considered U.S. Patent Nos. 9,143,561 (“’561 patent”) [EX1001]; 10,067,942 (“’942 patent”) [EX1002]; 10,289,607 (“’607 patent”) [EX1003]; 10,642,787 (“’787 patent”) [EX1004]; 10,754,823 (“’823 patent”) [EX1005]; and 11,003,622 (“’622 patent”) [EX1006] (collectively, “Challenged Patents”) and the respective prosecution histories of the Challenged Patents. I have cited to the following documents in my analysis below:

Ex. No.	Description of Document
1001	U.S. Patent No. 9,143,561 (“’561” or “’561 patent”)
1002	U.S. Patent No. 10,067,942 (“’942” or “’942 patent”)
1003	U.S. Patent No. 10,289,607 (“’607” or “’607 patent”)
1004	U.S. Patent No. 10,642,787 (“’787” or “’787 patent”)
1005	U.S. Patent No. 10,754,823 (“’823” or “’823 patent”)
1006	U.S. Patent No. 11,003,622 (“’622” or “’622 patent”)
1007	Prosecution History of the ’561 Patent
1008	Prosecution History of the ’942 Patent
1009	Prosecution History of the ’607 Patent

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

Ex. No.	Description of Document
1010	Prosecution History of the '787 Patent
1011	Prosecution History of the '823 Patent
1012	Prosecution History of the '622 Patent
1013	Case No. IPR2022-00782, Paper No. 1, Petition for IPR
1014	Case No. IPR2022-00782, Paper No. 17, Decision Granting Institution of IPR (Redacted – Public Version)
1015	U.S. Provisional Patent Application No. 60/986,896 (“’896 provisional”)
1016	U.S. Patent Application Publication No. 2007/0174246 A1 to Johann Tomas Sigurdsson, et al. (“Sigurdsson”)
1017	File History of U.S. Patent Application Publication No. 2007/0174246 A1 to Johann Tomas Sigurdsson, et al. (“Sigurdsson File History”)
1018	U.S. Patent No. 7,734,690 B2 to George P. Moromisato, et al. (“Moromisato”)
1019	U.S. Patent Application Publication No. 2006/0190506 A1 to Rajesh M. Rao, et al. (“Rao”)
1020	U.S. Patent Application Publication No. 2005/0091289 A1 to Michael Shappell, et al. (“Shappell”)
1021	Microsoft Computer Dictionary (5th ed. 2002) (excerpts)
1022	Jim Boyce, <i>Microsoft Outlook Version 2002 Inside Out</i> (2001) (“Boyce”) (excerpts)
1023	European Patent Application Publication No. EP 1,621,990 A2 to Kitamaru, et al. (“Kitamaru”)
1024	U.S. Patent No. 5,923,325 to Ronald Jason Barber and Edwin Joseph

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Ex. No.	Description of Document
	Selker (“Barber”)
1025	U.S. Patent No. 7,035,847 B2 to David K. Brown, et al. (“Brown”)
1026	U.S. Patent Application Publication No. 2005/0120082 A1 to Lambertus Hesselink, et al. (“Hesselink”)

II. PERSON OF ORDINARY SKILL IN THE ART (“POSA”)

15. I understand that an assessment of claims of the Challenged Patents should be undertaken from the perspective of a person of ordinary skill in the art as of the earliest claimed priority date, which I understand is November 10, 2008 for all of the Challenged Patents.¹ If the priority date of the Challenged Patents were determined to be earlier (for example, November 9, 2007—the filing date of U.S. Provisional Patent Application No. 60/986,896—or May 18, 2004—the date that I understand the Patent Owner to be asserting in district court), my opinions expressed herein regarding the appropriate level of a person having ordinary skill in the art (“POSA”) would not change.

16. I have been advised that to determine the appropriate level of skill of a POSA, the following factors may be considered: (1) the types of problems

¹ A more detailed discussion of the priority to which the Challenged Patents are entitled appears below in Section V.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

encountered by those working in the field and prior art solutions thereto; (2) the sophistication of the technology in question, and the rapidity with which innovations occur in the field; (3) the educational level of active workers in the field; and (4) the educational level of the inventor(s).

17. The Challenged Patents generally relate to systems and methods for the management of digital files across distributed networks (*see, e.g.*, '561 patent, (54), Abstract); the prioritization of sending metadata of updated files before sending the updated files themselves (*see, e.g.*, '787 patent, (54), Abstract); and the display of indicators based on prioritized metadata indicating availability to download updated files (*see, e.g.*, '823 patent, (54), Abstract). In my opinion, a POSA as of November 2008 would have possessed a bachelor's degree in computer science (or similar degree), and two years of work experience developing software, including at least some experience working with data storage, data synchronization, and human-computer interaction technologies. A person could also have qualified as a POSA with some combination of (1) more formal education (such as a master's of science degree) and less technical experience, or (2) less formal education and more technical or professional experience in the fields listed above.

18. My opinions regarding the level of ordinary skill in the art are based on, among other things, my more than 30 years of experience in computer science,

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

my understanding of the basic qualifications that would be relevant to an engineer or scientist tasked with investigating methods and systems in the relevant technical area, and my familiarity with the backgrounds of colleagues, co-workers, students, and employees, both past and present.

19. I have been informed that Unified Patents took the following position regarding the level of skill of a POSA in an IPR challenging the '942 patent: "A person of ordinary skill in the art ('POSA') would have been one having, by November 9, 2007: (1) an undergraduate degree in computer science, computer engineering, or closely-related field, or similar advanced post-graduate education; and (2) two or more years of experience with file sharing systems, with more experience substituting for less education and vice versa." EX1013, Case No. IPR2022-00782, Paper No. 1, Petition for IPR at 6. I view this articulation of the level of skill of a POSA to be quite similar to the level of skill I articulated above. My opinions in this declaration would not change based on the POSA being assumed to have the level of skill proposed by Unified Patents.

20. As of May 18, 2004 and at all times thereafter, my qualifications and experience have exceeded those of the hypothetical POSA I proposed above and the hypothetical POSA proposed in the *Unified Patents* matter. Although my analysis and opinions regarding the Challenged Patents have been based on the perspective

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

of a POSA as of November 10, 2008, my analysis and opinions would not change based on the perspective of a POSA as of November 9, 2007 or May 18, 2004.

21. My opinions herein regarding the understanding of a POSA and my other opinions set forth herein would remain the same if the POSA were determined to have somewhat more or less education and/or experience than I have identified above.

III. STATEMENT OF LEGAL PRINCIPLES

A. Claim Construction

22. The following sets forth my understanding of the legal principles of claim construction from counsel. I understand that under the legal principles, claim terms are generally given their ordinary and customary meaning, which is the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention, i.e., as of the effective filing date of the patent application. I further understand that the person of ordinary skill in the art is deemed to read the claim term not only in the context of the particular claim in which a claim term appears, but in the context of the entire patent, including the specification.

23. I understand by counsel that the patent specification, under the legal principles, has been described as the single best guide to the meaning of a claim term, and is thus highly relevant to the interpretation of claim terms. And I

understand for terms that do not have a customary meaning within the art, the specification usually supplies the best context of understanding the meaning of those terms.

24. I am further informed by counsel that other claims of the patent in question, both asserted and unasserted, can be valuable sources of information as to the meaning of a claim term. Because the claim terms are normally used consistently throughout the patent, the usage of a term in one claim can often illuminate the meaning of the same term in other claims. Differences among claims can also be a useful guide in understanding the meaning of particular claim terms.

25. I understand that the prosecution history can further inform the meaning of the claim language by demonstrating how the inventors understood the invention and whether the inventors limited the invention in the course of prosecution, making the claim scope narrower than it otherwise would be. Extrinsic evidence may also be consulted in construing the claim terms, such as my expert testimony.

26. I have been informed by counsel that, in *Inter Partes* Review (IPR) proceedings, a claim of a patent shall be construed using the same claim construction standard that would be used to construe the claim in a civil action filed in a U.S. district court (which I understand is called the “*Phillips*” claim construction standard), including construing the claim in accordance with the ordinary and

customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.

27. I have been instructed by counsel to apply the “*Phillips*” claim construction standard for purposes of interpreting the claims in this proceeding, to the extent they require an explicit construction. The description of the legal principles set forth above thus provides my understanding of the “*Phillips*” standard as provided to me by counsel.

B. Obviousness

28. The following sets forth my understanding of the legal principles of obviousness from counsel. I understand that a patent claim is obvious if, as of the critical date (i.e., either the earliest claimed priority date (pre-AIA) or the effective filing date (AIA)), it would have been obvious to a person having ordinary skill in the field of the technology (the “art”) to which the claimed subject matter belongs.

29. I understand that the following factors should be considered in analyzing obviousness: (1) the scope and content of the prior art; (2) the differences between the prior art and the claims; and (3) the level of ordinary skill in the pertinent art. I also understand that certain other facts known as “secondary considerations” such as commercial success, unexplained results, long felt but unsolved need, industry acclaim, simultaneous invention, copying by others, skepticism by experts

in the field, and failure of others may be utilized as indicia of nonobviousness. I understand, however, that secondary considerations should be connected, or have a “nexus,” with the invention claimed in the patent at issue.

30. I understand that a reference qualifies as prior art for obviousness purposes when it is analogous to the claimed invention. The test for determining what art is analogous is: (1) whether the art is from the same field of endeavor, regardless of the problem addressed, and (2) if the reference is not within the field of the inventor’s endeavor, whether the reference still is reasonably pertinent to the particular problem with which the inventor is involved.

31. I understand that a person of ordinary skill in the art is assumed to have knowledge of all prior art. I understand that one skilled in the art can combine various prior art references based on the teachings of those prior art references, the general knowledge present in the art, or common sense. I understand that a motivation to combine references may be implicit in the prior art, and there is no requirement that there be an actual or explicit teaching to combine two references. Thus, one may take into account the inferences and creative steps that a person of ordinary skill in the art would employ to combine the known elements in the prior art in the manner claimed by the patent at issue. I understand that one should avoid “hindsight bias” and ex post reasoning in performing an obviousness analysis. But

this does not mean that a person of ordinary skill in the art for purposes of the obviousness inquiry does not have recourse to common sense.

32. I understand that when determining whether a patent claim is obvious in light of the prior art, neither the particular motivation for the patent nor the stated purpose of the patentee is controlling. The primary inquiry has to do with the objective reach of the claims, and that, if those claims extend to something that is obvious, then the entire patent claim is invalid.

33. I understand one way that a patent can be found obvious is if there existed at the time of the invention a known problem for which there was an obvious solution encompassed by the patent's claims. I understand that a motivation to combine various prior art references to solve a particular problem may come from a variety of sources, including market demand or scientific literature. I understand that a need or problem known in the field at the time of the invention can also provide a reason to combine prior art references and render a patent claim invalid for obviousness. I understand that familiar items may have obvious uses beyond their primary purpose, and that a person of ordinary skill in the art will be able to fit the teachings of multiple prior art references together like the pieces of a puzzle. I understand that a person of ordinary skill is also a person of at least ordinary creativity. I understand when there is a design need or market pressure to solve a

problem and there are a finite number of identified, predictable solutions, a person of ordinary skill has good reason to pursue the known options within his or her technical grasp. If this finite number of predictable solutions leads to the anticipated success, I understand that the invention is likely the product of ordinary skill and common sense, and not of any sort of innovation. I understand that the fact that a combination was obvious to try might also show that it was obvious, and hence invalid, under the patent laws. I understand that if a patent claims a combination of familiar elements according to known methods, the combination is likely to be obvious when it does not more than yield predictable results. Thus, if a person of ordinary skill in the art can implement a predictable variation, an invention is likely obvious. I understand that combining embodiments disclosed near each other in a prior art reference would not ordinarily require a leap of inventiveness.

34. I understand that obviousness may be shown by demonstrating that it would have been obvious to modify what is taught in a single piece of prior art to create the patented invention. Obviousness may also be shown by demonstrating that it would have been obvious to combine the teachings of more than one item of prior art. I understand that a claimed invention may be obvious if some teaching, suggestion, or motivation exists that would have led a person of ordinary skill in the art to combine the invalidating references. I also understand that this suggestion or

motivation may come from the knowledge of a person having ordinary skill in the art, or from sources such as explicit statements in the prior art. I understand that when there is a design need or market pressure, and there are a finite number of predictable solutions, a person of ordinary skill may be motivated to apply common sense and his skill to combine the known options in order to solve the problem.

35. I understand the following are examples of approaches and rationales that may be considered in determining whether a piece of prior art could have been combined with other prior art or with other information within the knowledge of a person having ordinary skill in the art:

- (1) Some teaching, motivation, or suggestion in the prior art that would have led a person of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention;
- (2) Known work in one field of endeavor may prompt variations of it for use in the same field or a different field based on design incentives or other market forces if the variations would have been predictable to a person of ordinary skill in the art;
- (3) Combining prior art elements according to known methods to yield predictable results;
- (4) Applying a known technique to a known device, method, or product ready for improvement to yield predictable results;
- (5) Applying a technique or approach that would have been “obvious

to try” (choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success);

(6) Simple substitution of one known element for another to obtain predictable results; or

(7) Use of a known technique to improve similar products, devices, or methods in the same way.

36. I understand that, when determining whether a claimed combination is obvious, the correct analysis is not whether one of ordinary skill in the art, writing on a blank slate, would have chosen the particular combination of elements described in the claim. Instead, I understand the correct analysis considers whether one of ordinary skill, facing the wide range of needs created by developments in the field of endeavor, would have seen a benefit to selecting the combination claimed.

37. I understand that the test for obviousness is not whether the features of a secondary reference may be bodily incorporated into the structure of the primary reference. The test for obviousness, in other words, is not whether the references could be physically combined but whether the claimed inventions are rendered obvious by the teachings of the prior art as a whole.

IV. OVERVIEW OF THE CHALLENGED PATENTS

38. In applying and analyzing the claim language of the Challenged Claims, I have considered and interpreted the Challenged Claims in light of the claim language of all of the Challenged Patents, the specifications of the patents, and the

patents' file histories.

A. The Specifications of the Challenged Patents

1. Each of the Challenged Patents Recites a Largely Identical Specification

39. I note that the specifications of the Challenged Patents are almost identical, with the exception of the Summary of Invention sections that largely track the language of the respective claims of the Challenged Patents. Thus, I summarize here the '823 patent's specification as an exemplary specification of the Challenged Patents. The '823 patent generally relates to “computer-implemented processes and, more specifically, to sharing of electronic files among computer systems.” ('823, 1:24–26).

40. The '823 patent states in the “Background” section that “customers are increasingly using more than one computing device” and that “[a]lthough modern devices are easily connected, they do not provide the customer a seamless environment” to manage files across devices. ('823, 1:34–44). The patent asserts that “several solutions” including Remote Desktop software, Distributed File Systems, FTP, E-mail, Flash Drives, and External Disk Drives are “in one way or another inadequate to the task.” ('823, 2:18–3:47).

41. The patent purports to address the perceived deficiencies of the prior art through “automatic modification-triggered transfer of a file among two or more

computer systems associated with a user.” (’823, 3:55-57). In the “Summary of the Invention” section, the patent states that “responsive to a user modifying a content of the file at a first client device (associated with the user), a server system may automatically receive a copy of the file from the first client device, where the file copy may be an updated version of the file that is generated from the user modifying the content of the file. After receiving metadata associated with the updated version of the file from the first client device, the server system may automatically transfer the metadata to a second client device associated with the user such that, before the file copy is transferred to the second client device, the transfer of the metadata to the second client device causes a graphical availability indication of the updated version of the file to be presented (e.g., proximate a file icon representing the file) at the second client device based on the metadata.” (’823, 4:3-18).

42. The patent discloses a computing system environment that may be comprised of “personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and the like.” (’823, 5:14–25). The user’s device may “operate in a networked environment using logical connections to one or more remote

computers, such as a remote computer” which “may be a personal computer, a server, a router, a network PC, a peer device or other common network node.” (’823, 7:13–17). The user’s devices may be “linked via the network 220 to server 230 so that computer programs, such as, for example, a browser or other applications, running on one or more of the user devices 210, 280 can cooperate in two-way communication with server 230 and one or more applications running on server 230.” (’823, 7:65–8:3).

43. In an embodiment of the invention allowing a user “to automatically maintain synchronization” between the user’s devices, the “server 230 or other appropriate electronic device transfers a file-transfer and/or synchronization application to the user devices 210, 280 for installation thereon.” (’823, 8:9–18). The file-transfer application “provides file-transfer clients 310, 320 executable by the user devices” as well as “user interfaces 350, 360 that are displayable on the displays 250, 290, respectively.” (’823, 8:18–29). These interfaces “allow a user to view, access and/or organize files to be synched among the various user devices.” (’823, 8:29–31).

44. In this embodiment, “all files that the user desires to be synched” will be sent to the server which can “store such files in the [server’s] storage 270 and/or transfer the files to one or more of the respective hard drives of the user devices 210,

280, thereby enabling each respective user device to access such files.” (’823, 8:32–39). The patent states that the server will typically store the files “only if the destination user device is offline or otherwise temporarily not in communication with the server 230” and “will transfer the temporarily stored files to the destination device” upon resuming connection. (’823, 8:42–47).

45. In another embodiment, the files or copies of the files are transferred to the server “[u]pon a predetermined and user-configurable triggering event” that may include “the user saving the file, the elapsing of a predetermined amount of time during which the file has been opened, or the re-initiation of a communication session between the device 210 and the server 230.” (’823, 8:53–60).

46. The patent discloses that the server that is executing “a synchronization application 380” that “monitors a set of user devices to which the file 370 should be transferred to” and polls these devices to determine whether the devices are in communication with the server. (’823, 8:61–9:2). If the sync application determines that a device is in communication with the server, then it “transfers the file 370 to the device 280, whereupon the transfer client 320 resident on the device 280 replaces the previous version of the file 370, previously cached on the device 280, with the latest version of the file 370.” (’823, 9:3–8). If the sync application determines that a device is not in communication with the server, it “may store the file 370 in the

storage 270 until such time as communication between the device 280 and server 230 is reestablished.” (’823, 9:8–12).

47. In an embodiment, the user interfaces may “include a list of the customer's documents and related metadata” and “provide customers with an accurate ‘picture’ of their document collection, regardless of whether their devices physically contain the documents.” (’823, 9:17–23). The user interface may include “[a]n indicator signifying a document's availability may be prominently displayed adjacent to the document's representation so that customers are aware of the document's availability.” (’823, 9:32–35). This “directory update system updates at a higher priority than the documents to be synchronized.” (’823, 9:28–29).

48. In another embodiment, a stand-alone application allows users “to find and manage documents associated with transfer clients 310, 320 by visualizing relationships between documents and their metadata.” (’823, 9:36–39). Users can organize documents based on different parameters including application, most recent access, project/sub-project, people, and process stage. (’823, 9:53–10:11).

49. The patent states that an embodiment prevents broken links by “(1) always ensuring the file catalog (e.g., the stand-alone application that allows customers to find and manage documents associated with transfer clients 310, 320, as discussed above herein) is synchronized before any file data is synchronized, and

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

(2) pausing any file access by any program until the file contents have been fully
synchronized.” (’823, 10:26–42).

2. Summary of the Specification from the *Unified Patents* IPR

50. I have been informed that Unified Patents filed an IPR challenging
certain claims of the ’942 patent. (See EX1014, *Unified Pats., LLC v. Topia Tech.,
Inc.*, IPR2022-00782, Paper 17 (P.T.A.B. Nov. 17, 2022) (“*Unified Pats.*”).) I
understand that the Board provided this summary of the ’942 patent, which I agree
with and have applied in this Declaration as a representative description of the
mostly common specification shared by the Challenged Patents:

The ’942 patent is titled “Architecture for Management of Digital
Files Across Distributed Network.” Ex. 1001, code (54). The ’942
patent concerns sharing files between electronic devices and aims to
improve file transfer and synchronization between a user’s multiple
computing devices. *Id.*, 1:18-36, 4:49-54. Figure 2 of the ’942 patent
shows an embodiment of the invention and is reproduced below.

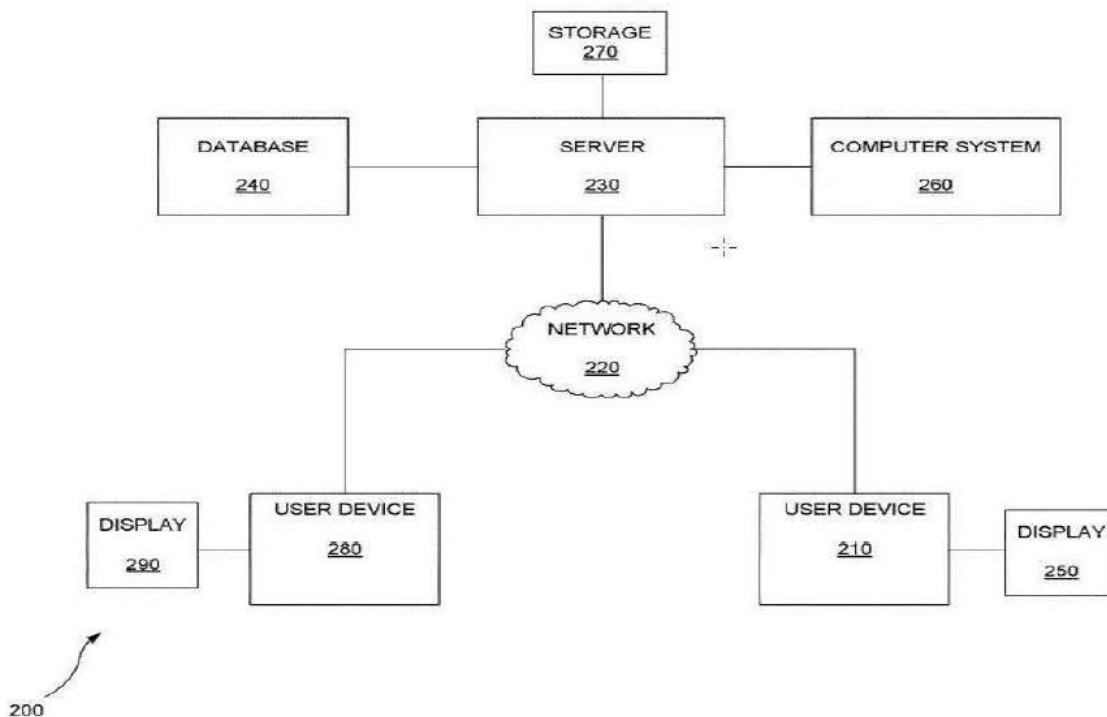


FIG. 2

Figure 2 shows computer network system 200 including electronic user devices 210, 280, such as personal computers or workstations, that are linked via network 220 (e.g., the Internet) to an electronic device or system, such as server 230. '942, 7:26-32. Applications running on one or more user devices 210, 280 can cooperate in two-way communication with server 230 and one or more applications running on server 230. *Id.*, 7:49-54. A file transfer and/or synchronization application on user devices 210, 280 having a user interface allows a user to view, access, and/or organize files to be synched among the user devices. *Id.*, 7:65-8:15.

A file that the user desires to be synched or shared is uploaded by user device 210 and received by server 230. *Id.*, 8:16-18, 8:32-44. Server 230 stores the received file to storage 270 if destination user device 280 is offline or otherwise temporarily not in communication with server 230. *Id.*, 8:26-29, 8:45-63. Upon resuming communication with destination user device 280, server 230 transfers the temporarily

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

stored file to user device 280, whereupon the previous version of the file on user device 280 is replaced with the latest version of the file modified on user device 210. *Id.*, 8:29-31, 8:51-59. A similar reverse-direction synchronization process may be performed with regard to a file modified on user device 280 and synchronized to user device 210. *Id.*, 8:63-67.

Unified Pats., 7-8.

B. File Histories of the Challenged Patents

51. I have reviewed the file histories of the Challenged Patents and, in my opinion, the prior art combinations that I rely upon in this Declaration are not cumulative of the prior art combinations considered during prosecution of any of the Challenged Patents. In addition, with the exception of the Sigurdsson reference, which was considered during prosecution of the '622 patent, none of the prior art I rely upon in this Declaration is cumulative of the prior art cited on the face of the patents. In addition, for the reasons below, the prior art combinations I rely upon herein disclose and/or render obvious the limitations that the USPTO determined to allegedly be missing in the prior art during prosecution of the Challenged Patents.

1. '823 Patent's File History

52. I understand from my review of the '823 patent's file history that the claims that issued in the '823 patent were allowed in view of an examiner's amendment that added the following claim limitations:

subsequent to the transfer of the first metadata to the second client

device, transfer the copy of the first file to the second client device,

wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration

EX1011, '823 File History, Notice of Allowance at 5-13. For the reasons discussed below, it is my opinion that the prior art combinations relied upon in this Declaration disclose and/or render obvious these limitations. Based on my review of the file history, it appears that neither the combinations relied upon herein nor similar combinations were before the examiner during prosecution.

2. '787 Patent's File History

53. The examiner stated that the '787 patent's claims were patentable over the prior art because, according to the examiner, the prior art did not disclose, make obvious, or otherwise suggest" the following: "wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration." EX1010, '787 File History, Notice of Allowance at 3. As discussed above in connection with the '823 patent's file

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

history, it is my opinion that the prior art combinations relied upon in this Declaration disclose and/or render obvious this limitation. Based on my review of the file history, it appears that neither the combinations relied upon herein nor similar combinations were before the examiner during prosecution.

3. '607 Patent's File History

54. The examiner allowed the claims of the '607 patent to issue in view of claim amendments that added, *inter alia*, limitations related to assigning a higher transmission priority to metadata than file data. See EX1009, '607 File History, Notice of Allowance at 3-18. It is my opinion that the prior art combinations relied upon in this Declaration disclose and/or render obvious this limitation as well as the other limitations that were added prior to allowance of the claims. Based on my review of the file history, it appears that neither the combinations relied upon herein nor similar combinations were before the examiner during prosecution.

4. '942 Patent's File History

55. The examiner determined that the following limitations added to the '942 patent's claims during prosecution were not disclosed in the prior art before the examiner:

automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

receiving the modified second electronic file copy from the third
electronic device

wherein, responsive to sending the modified second electronic file copy
to the second electronic device, an older version of the second
electronic file stored on the second electronic device is automatically
caused to be replaced with the modified second electronic file copy
such that the modified second electronic file copy is stored on the
second electronic device in lieu of the older version of the second
electronic file

EX1008, '942 File History, Notice of Allowability at 12. For the reasons discussed
below, it is my opinion that the prior art combinations relied upon in this Declaration
disclose and/or render obvious these limitations. Based on my review of the file
history, it appears that neither the combinations relied upon herein nor similar
combinations were before the examiner during prosecution.

5. '561 Patent's File History

56. The examiner determined that the following limitations added to the
'561 patent's claims during prosecution were not disclosed in the prior art before the
examiner:

automatically transfer the modified second electronic file copy to the
second electronic device to replace an older version of the second
electronic file stored on the second electronic device with the modified
second electronic file copy having the content modified by the user;

wherein the second application automatically transfers the copy of the
modified first electronic file to the first electronic device upon
determining that a save operation has been performed on the modified
first electronic file

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

See EX1007, '561 File History, Notice of Allowance at 3-18. For the reasons discussed below, it is my opinion that the prior art combinations relied upon in this Declaration disclose and/or render obvious these limitations. Based on my review of the file history, it appears that neither the combinations relied upon herein nor similar combinations were before the examiner during prosecution.

6. '622 Patent's File History

57. Upon review of the file history of the '622 patent, I observed the examiner's citation and discussion of the Sigurdsson reference in rejecting versions of the claims of the '622 patent that I understand were subsequently amended to overcome rejections in view of Sigurdsson in combination with other references. EX1012, File History of '622 Patent, November 10, 2020 Non-Final Rejection at 4-21. Specifically, I understand that the applicant overcame such rejections by amending the claims of the '622 patent to recite the following: "receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file" and "automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

file being transferred to the second client device.” EX1012, File History of ’622
Patent, December 14, 2020 Claim Amendments at 2, 5, 7.

58. For the reasons discuss below (see ¶¶ 92-102, 394-411), I believe that these limitations added to the claims of the ’622 patent would have been obvious in view of Sigurdsson in combination with the Rao and/or Shappell references. I understand that Rao and Shappell were not cited or discussed by the examiner or the applicant during prosecution of the ’622 patent. In addition, I have reviewed the prior art references that were cited during prosecution of the application that issued as the ’622 patent and concluded that, unlike the Rao and Shappell references, they do not expressly disclose these limitations. Thus, a POSA would have viewed Rao and Shappell to be significantly different in kind from the prior art references cited on the face of the ’622 patent, in that Rao and Shappell expressly disclose assigning relatively higher priorities to the transmission of file metadata than file data. In my opinion, therefore, the combinations of Sigurdsson with Rao and/or Shappell are not “substantially the same” as the prior art or prior art combinations considered by the examiner during prosecution of the ’622 patent.

C. The Challenged Claims of the Challenged Patents

1. The Challenged Claims of the ’823 Patent

59. I understand that Petitioner argues that the following claims of the ’823

patent are invalid as obvious: claims 1-4, 8-11, and 13-16. These claims recite as follows:

[1p]² A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1d] automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[1e] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and

² I have added these labels for each element of the Challenged Claims and refer to them throughout my report.

[1f] (ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and

[1g] wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

[1h] subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,

[1i] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[2] The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

[3p] The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:

[3a] receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

[3b] determine that the server system is in communication with the first client device associated with the user; and

[3c] automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

[4] The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

[8p] A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:

[8a] receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[8b] receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[8c] automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[8d] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version

of the first file to be presented at the second client device based on the first metadata, and

[8e] (ii) the graphical availability indication is presented proximate a graphical file representation of the first file on a user interface of the second client device, and

[8f] wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

[8g] subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device,

[8h] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[9] The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

[10p] The method of claim 8, further comprising:

[10a] receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

[10b] determining that the server system is in communication with the first client device associated with the user; and

[10c] automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

[11] The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

[13p] One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:

[13a] receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[13b] receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[13c] automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[13d] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and

[13e] (ii) the availability indication is presented proximate a graphical file representation of the first file on a user interface of the second client device, and

[13f] wherein the availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

[13g] subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device,

[13h] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[14] The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

[15p] The media of claim 13, the operations further comprising:

[15a] receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the

second file that is generated from the user modifying the content of the second file;

[15b] determining that the server system is in communication with the first client device associated with the user; and

[15c] automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

[16] The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

2. The Challenged Claims of the '787 Patent

60. I understand that Petitioner argues that the following claims of the '787 patent are invalid as obvious: claims 1-4, 8-11, and 13-16. These claims recite as follows:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

[1d] automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[1e] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and

[1f] (ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

[1g] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[2] The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

[3p] The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:

[3a] receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

[3b] determine that the server system is in communication with the first client device associated with the user; and

[3c] automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

[4] The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

[8p] A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:

[8a] receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[8b] receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is

generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

[8c] automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[8d] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and

[8e] (ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

[8f] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[9] The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

[10p] The method of claim 8, further comprising:

[10a] receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the

user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

[10b] determining that the server system is in communication with the first client device associated with the user; and

[10c] automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

[11] The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

[13p] One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:

[13a] receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[13b] receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

[13c] automatically transferring, based on the first priority being greater

than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[13d] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and

[13e] (ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

[13f] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[14] The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

[15p] The media of claim 13, the operations further comprising:

[15a] receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;

[15b] determining that the server system is in communication with the first client device associated with the user; and

[15c] automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

[16] The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

3. The Challenged Claims of the '607 Patent

61. I understand that Petitioner argues that the following claims of the '607 patent are invalid as obvious: claims 1, 3-5, 7, 12, 14, 17, 19, and 20. These claims recite as follows:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the

content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1d] determine that the server system is not in communication with a second client device associated with the user;

[1e] store the copy of the first file on the server system;

[1f] automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[1g] automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

[3] The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[4] The system of claim 1, wherein availability of the version of the first file is presented at the second client device based on the first metadata.

[5p] The system of claim 1, wherein the server system is caused to:

[5a] receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being a version of the second file

that is generated from the user modifying the content of the second file;

[5b] determine that the server system is in communication with the first client device associated with the user; and

[5c] automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

[7] The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

[12p] A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:

[12a] receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[12b] receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[12c] determining, by the server system, that the server system is not in communication with a second client device associated with the user;

[12d] store the copy of the first file on the server system;

[12e] automatically transferring, by the server system, the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[12f] automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

[14] The method of claim 12, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[17p] A system comprising:

[17a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[17b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[17c] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority

greater than a second priority assigned to the copy of the first file, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration;

[17d] determine that the server system is not in communication with a second client device associated with the user;

[17e] store the copy of the first file on the server system;

[17f] automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[17g] automatically transfer, over a network, the copy of the first file to the second client device associated with the user, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device,

[17h] wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.

[19p] One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:

[19a] receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the

first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[19b] receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[19c] determining, by the server system, that the server system is not in communication with a second client device associated with the user;

[19d] storing, by the server system, the copy of the first file on the server system;

[19e] automatically transferring, by the server system, the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[19f] automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

[20] The one or more non-transitory machine-readable media of claim 19, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

4. The Challenged Claims of the '622 Patent

62. I understand that Petitioner argues that the following claims of the '622 patent are invalid as obvious: claims 1, 3-5, 7, 11, 12, 14, 16, and 17. These claims recite as follows:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[1c] store the copy of the first file on the server system;

[1d] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1e] automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[1f] automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

[3] The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[4] The system of claim 1, wherein availability of the version of the first file is presented at the second client device based on the first metadata.

[5p] The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:

[5a] receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being a version of the second file that is generated from the user modifying the content of the second file;

[5b] determine that the server system is in communication with the first client device associated with the user; and

[5c] automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the first file from the second client device.

[7] The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

[11p] A method being implemented by a server system comprising one or more processors executing computer program instructions that, when

executed, perform the method, the method comprising:

[11a] receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[11b] receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[11c] automatically transferring, by the server system, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[11d] automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

[12p] The method of claim 11, further comprising:

[12a] initially determining, by the server system, that the server system is not in communication with the second client device associated with the user; and

[12b] automatically transferring, by the server system, the copy of the first file to the second client device to replace the older version of the first file stored on the second client device with the copy of the first file, responsive to (i) resuming communication with the second client

device, (ii) determining that the server system is in communication with the second client device, (iii) and receiving the copy of the first file from the first client device.

[14] The method of claim 11, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

[16p] One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a first computer system, cause operations comprising:

[16a] receiving, by the first computer system, over a network, a copy of a first file from a second computer system associated with a user, wherein the copy of the first file is automatically received from the second computer system responsive to the user modifying a content of the first file stored on the second computer system, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[16b] storing, by the first computer system, the copy of the first file on the first computer system;

[16c] receiving, by the first computer system, from the second computer system, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[16d] automatically transferring, by the first computer system, based on the first priority being greater than the second priority, the first metadata to a third computer system associated with the user such that the first metadata is transferred to the third computer system prior to the copy of the first file being transferred to the third computer system;

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

and

[16e] automatically transferring, by the first computer system, over a network, the copy of the first file to the third computer system associated with the user to replace an older version of the first file stored on the third computer system, responsive to receiving the copy of the first file from the second computer system.

[17] The one or more non-transitory machine-readable media of claim 16, wherein at least one of the first computer system or the second computer system comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the first computer system or the second computer system assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

5. The Challenged Claims of the '942 Patent

63. I understand that Petitioner argues that the following claims of the '942 patent are invalid as obvious: claims 1, 3, 4, 10, 12 and 13. These claims recite as follows:

[1p] A system comprising:

[1a] a first electronic device, associated with a user, configured to:

[1b] receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;

[1c] determine whether the first electronic device is in communication with a third electronic device;

[1d] automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;

[1e] receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;

[1f] determine whether the first electronic device is in communication with the second electronic device; and

[1g] automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,

[1h] wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and

[1i] wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second

electronic file.

[3] The system of claim 1, wherein the first electronic device is configured to: store, via the first application, the modified first electronic file copy to a memory device associated with the first electronic device.

[4] The system of claim 3, wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.

[10p] A method comprising:

[10a] receiving, via a first application at a first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with a user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;

[10b] determining whether the first electronic device is in communication with a third electronic device;

[10c] automatically sending, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;

[10d] receiving, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;

[10e] determining whether the first electronic device is in communication with the second electronic device; and

[10f] automatically sending, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,

[10g] wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and

[10h] wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

[12] The method of claim 10, further comprising: storing, via the first application, the modified first electronic file copy to a memory device associated with the first electronic device.

[13] The method of claim 12, wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.

6. The Challenged Claims of the '561 Patent

64. I understand that Petitioner argues that the following claims of the '561 patent are invalid as obvious: claims 1, 3, 4, 8, 10 and 11. These claims recite as follows:

[1p] A system, comprising:

[1a] a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:

[1b] receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and

[1c] wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and

[1d] wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and

[1e] automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;

[1f] wherein the second application automatically transfers the copy of

the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.

[3] The system of claim 1 wherein the first application is configured to store the modified first electronic file copy to a memory device associated with the first electronic device.

[4] The system of claim 3, wherein the first application is configured to store the modified first electronic file copy to the memory device associated with the first electronic device when the third electronic device is not in communication with the first electronic device.

[8p] A method implementable in an electronic system having a storage component, the electronic system being coupled to a first electronic device having stored thereon a first electronic file and a second electronic device having stored thereon a second electronic file, the first electronic device and the second electronic device each being associated with a user, the method comprising:

[8a] receiving from the first electronic device a copy of a first electronic file modified by the user, the copy of the modified first electronic file being automatically provided to the electronic system by the first electronic device when a content of the first electronic file is modified by the user;

[8b] receiving from the second electronic device a copy of a second electronic file modified by the user, the copy of the modified second electronic file being automatically provided to the electronic system by the second electronic device when a content of the second electronic file is modified by the user;

[8c] automatically transferring the modified first electronic file copy to the second electronic device to replace an older version of the first electronic file stored on the second electronic device with the modified first electronic file copy having the content modified by the user; and

[8d] automatically transferring the modified second electronic file copy

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

to the first electronic device to replace an older version of the second electronic file stored on the first electronic device with the modified second electronic file copy having the content modified by the user;

[8e] wherein the first electronic device automatically transfers the copy of the modified first electronic file to the electronic system upon determining that a save operation has been performed on the modified first electronic file.

[10] The method of claim 8, wherein the electronic system is configured to store the modified first electronic file copy to the storage component.

[11] The method of claim 10 wherein the electronic system is configured to store the modified first electronic file copy to the storage component when the second electronic device is not in communication with the electronic system.

V. THE CHALLENGED CLAIMS OF THE '823, '787, '607, '622, '942, AND '561 PATENTS ALL RECITE SUBJECT MATTER THAT IS NOT DISCLOSED IN U.S. PROVISIONAL APPLICATION NO. 60/986,896

65. I understand that all of the Challenged Patents claim priority to U.S. Provisional Application No. 60/986,896 (“’896 provisional”) (EX1015), filed on November 9, 2007. I have been asked to provide an opinion as to whether a POSA would have understood the ’896 provisional to disclose the subject matter claimed in any of the Challenged Claims. Based on my review of the ’896 provisional, it is my opinion that each of the Challenged Claims recites one or more limitations that are not disclosed in the ’896 provisional. For this reason, I have been instructed to assume a priority date of no earlier than November 10, 2008 for each of the Challenged Patents.

A. The Challenged Claims of the '823 Patent Each Recite One or More Limitations that are Not Disclosed in the '896 Provisional

66. The Challenged Claims of the '823 patent each recite one or more limitations that are not disclosed in the '896 provisional.

1. “wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device”

67. Each Challenged Claim of the '823 patent recites this limitation, in substantially identical form. This limitation is not disclosed in the '896 provisional. The '896 provisional says nothing about automatically receiving a file from a first client device responsive to the user modifying content of the file. The '896 provisional discusses automated backups in the context of prior art systems, but these processes are not described as backing up files in response to a user modifying content of any file. '896 provisional, 15. Moreover, this discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

68. The '896 provisional also discusses prior art distributed file systems that could “automatically cach[e] often-used files on the local file system” and “only synchroniz[e] them when they have been changed.” '896 provisional, 14. However,

this excerpt describes receiving a file **at a client device** that might be a copy of the file (i.e., cached) that is also stored remotely, not receiving a file **from** a client device in response to a user modifying the file.³ *Id.* (“Additionally, **accessing a file located on a distributed file system** tends to be slower than accessing files on the local hard drive. . . . [T]his separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if **the file he is remotely accessing** is the latest version of the file, versus an earlier one that has been marked to be updated.”).⁴ And, again, this discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

³ Emphasis added throughout, unless otherwise noted.

⁴ For similar reasons, the limitation in claims 3, 10, and 15 reciting: “receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device” or substantially similar language are not disclosed in the ’896 provisional.

2. **“receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file” and “automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device”**

69. Each Challenged Claim of the '823 patent recites these limitations, in substantially identical form. The '896 provisional does not disclose receiving metadata from a client that has already been assigned a “first priority greater than a second priority” assigned to a copy of the file associated with the metadata. Although the '896 provisional discusses a “directory update system” that “updates at higher priority than the documents to be synchronized,” it discusses this directory update system in the context of updating MediaSpace’s “single logical area for customers to store and access their documents.” '896 provisional, 16-17. The '896 provisional does not describe the client as generating or transferring any metadata or assigning it any priority. In addition, the '896 provisional does not disclose **automatic** transfer of first metadata from a server to a “second client device” after the server receives the first metadata from a first client device—to the contrary, the '896 provisional suggests that such a transfer need only take place **manually** “when

a customer browses or searches through his set of documents.” To the extent the ’896 provisional discloses sending metadata at all, it discloses it in the context of MediaSpace’s directory update system ensuring that a customer may see a file that “ha[s] not yet been cached locally on the device,” which does not require use of metadata received from the client on which the file was modified. Thus, these limitations are not disclosed in the ’896 provisional.

3. **“wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented at the second client device based on the first metadata”**

70. Each Challenged Claim of the ’823 patent recites this limitation, in substantially identical form. This limitation is not disclosed in the ’896 provisional. Again, the ’896 provisional does not require the transfer of metadata from any client device before the sending of file data. The ’896 provisional describes: “An indicator signifying a document’s availability may be prominently displayed adjacent to the document’s representation in the MediaSpace so that customers are aware of the document’s availability.” ’896 provisional, 17-18. However, this indicator is not described as being generated from any metadata sent from a “first client device.” In other words, display of the indicator on a second client device may be caused by metadata created by a server running MediaSpace instead of being caused by

metadata created by a first client device.⁵

4. “determine that the server system is in communication with the first client device associated with the user”

71. Claims 3, 10, and 15 of the ’823 patent recite this limitation or substantially similar language. This limitation is not disclosed in the ’896 provisional. The ’896 provisional does not disclose any determination that any server is in communication with any particular client device. In fact, neither the word “communication” nor any of its related forms appears anywhere in the ’896 provisional.

5. “automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device”

72. Claims 3, 10, and 15 of the ’823 patent recite this limitation or substantially similar language. The ’896 provisional does not disclose this limitation. The ’896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not disclose the

⁵ For similar reasons, the limitation in claims 2, 9, and 14 reciting “before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata” or similar language are not disclosed in the ’896 provisional.

use of this functionality in the context of the alleged “MediaSpace” system or invention and does not disclose that synchronization must wait until communication between a server and client is detected. ’896 provisional, 14. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). Neither the word “replace” (nor any of its other forms) nor the concept of replacing an older file with a newer one appears anywhere in the ’896 provisional in the context of the allegedly inventive “MediaSpace” system. In addition, the ’896 provisional does not disclose receiving a copy of a file from one client device and using it to replace an older version of the same file on another client device.

6. **“wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device”**

73. Claims 4, 11, and 16 of the ’823 patent recite this limitation or substantially similar language. The ’896 provisional does not disclose this limitation. Neither the word “push” (or any of its other forms) nor the concept of sending a “push request” appears anywhere in the ’896 provisional. Although the ’896 provisional discusses a “Document Filer” that “allows a customer to file a

currently-opened document into the MediaSpace,” the Document Filer is not described as requiring pushing information to MediaSpace—the information could, for example, be pulled into MediaSpace.

B. The Challenged Claims of the ’787 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional

74. The Challenged Claims of the ’787 patent each recite one or more limitations that are not disclosed in the ’896 provisional.

2. “wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device”

75. Each of the Challenged Claims of the ’787 patent includes this limitation or a substantial identical limitation. For the reasons discussed above in connection with the same limitation recited in the ’823 patent’s Challenged Claims, this limitation is not disclosed in the ’896 provisional.⁶

⁶ For similar reasons, the limitation in claims 3, 10, and 15 of the ’787 patent reciting: “receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device” or substantially similar language are not disclosed in the ’896 provisional.

3. **“receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file” and “automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device”**

76. Each of the Challenged Claims of the '787 patent includes these limitations or substantial identical limitations. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.

4. **“wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented at the second client device based on the first metadata”**

77. Each of the Challenged Claims of the '787 patent includes this limitation or a substantial identical limitation. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.⁷

⁷ For similar reasons, the limitation in claims 2, 9, and 14 of the '787 patent reciting “before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the

5. **“the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device”**

78. Each of the Challenged Claims of the '787 patent includes this limitation or a substantial identical limitation. This limitation is not disclosed in the '896 provisional. The '896 provisional does not disclose any “file representation” that represents the updated version of any particular file that is stored on any particular client device but not on a different client device.

6. **“determine that the server system is in communication with the first client device associated with the user”**

79. Claims 3, 10, and 15 of the '787 patent recite this limitation or substantially similar language. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.

7. **“automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device”**

first file presented on the user interface of the second client device to be updated based on the first metadata” or similar language are not disclosed in the '896 provisional.

80. Claims 3, 10, and 15 of the '787 patent recite this limitation or substantially similar language. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.

8. **“wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device”**

81. Claims 4, 11, and 16 of the '787 patent recite this limitation or substantially similar language. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.

C. The Challenged Claims of the '607 Patent Each Recite One or More Limitations that are Not Disclosed in the '896 Provisional

82. The Challenged Claims of the '607 patent each recite one or more limitations that are not disclosed in the '896 provisional.

1. **“wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device”**

83. Each of the Challenged Claims of the '607 patent includes this limitation or a substantial identical limitation. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims,

this limitation is not disclosed in the '896 provisional.⁸

2. **“receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file” and “automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device”**

84. Each Challenged Claim of the '607 patent recites these limitations, in substantially identical form (claims 12 and 19 add that the “first metadata” is received “by the server system” and that “the server system” transfers “the first metadata to the second client device,” which are limitations that do not materially change the analysis). The '896 provisional does not disclose receiving metadata from a client that has already been assigned a “first priority greater than a second priority” assigned to a copy of the file associated with the metadata. Although the '896 provisional discusses a “directory update system” that “updates at higher priority than the documents to be synchronized,” it discusses this directory update

⁸ For similar reasons, the limitation in claim 5 of the '607 patent reciting: “receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device” or substantially similar language is not disclosed in the '896 provisional.

system in the context of updating MediaSpace’s “single logical area for customers to store and access their documents.” ’896 provisional, 16-17. The ’896 provisional does not describe the client as generating or transferring any metadata or assigning it any priority. In addition, the ’896 provisional does not disclose *automatic* transfer of first metadata from a server to a “second client device” after the server receives the first metadata from a first client device—to the contrary, the ’896 provisional suggests that such a transfer need only take place *manually* “when a customer browses or searches through his set of documents.” *Id.*, 17. To the extent the ’896 provisional discloses sending metadata at all, it discloses it in the context of MediaSpace’s directory update system ensuring that a customer may see a file that “ha[s] not yet been cached locally on the device,” which does not require use of metadata received from the client on which the file was modified. Thus, these limitations are not disclosed in the ’896 provisional.

3. “determine that the server system is not in communication with a second client device associated with the user”

85. Each Challenged Claim of the ’607 patent recites this limitation, in substantially identical form (claims 12 and 19 add that the determination is made “by the server system,” which does not materially change the analysis). This limitation is not disclosed in the ’896 provisional. The ’896 provisional does not disclose any determination that any server is or is not in communication with any

particular client device. In fact, neither the word “communication” nor any of its related forms appears anywhere in the ’896 provisional.

4. **“automatically transfer, over a network, the copy of the first file to the second client device associated with the user [to replace an older version of the first file stored on the second client device], responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device”**

86. Each Challenged Claim of the ’607 patent recites this limitation, in substantially identical form (claims 12 and 19 add that the automatic transferring is performed “by the server system,” which does not materially change the analysis). The ’896 provisional does not disclose this limitation. The ’896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not disclose the use of this functionality in the context of the alleged “MediaSpace” system or invention and does not disclose synchronization that is responsive to the resumption of communication between a server and client. ’896 provisional, 14. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). Neither the word “replace” (nor any of its other forms) nor the concept of replacing an older file with

a newer one appears anywhere in the '896 provisional in the context of the allegedly inventive "MediaSpace" system. In addition, the '896 provisional does not disclose receiving a copy of a file from one client device and using it to replace an older version of the same file on another client device.

5. **"wherein availability of the version of the first file is presented at the second client device based on the first metadata"**

87. Claim 4 of the '607 patent recites this limitation. This limitation is not disclosed in the '896 provisional. The '896 provisional does not disclose receiving metadata from a client that has already been assigned a "first priority greater than a second priority" assigned to a copy of the file associated with the metadata. Although the '896 provisional discusses a "directory update system" that "updates at higher priority than the documents to be synchronized," it discusses this directory update system in the context of updating MediaSpace's "single logical area for customers to store and access their documents." '896 provisional, 16-17. The '896 provisional does not disclose the presentation of availability of any version of any file based on metadata received from any client device. Instead, the presentation of availability of a version of a file in the MediaSpace system could be based on metadata generated at and sent from a MediaSpace server. In addition, the '896 provisional does not disclose **automatic** transfer of first metadata from a server to a

“second client device” after the server receives the first metadata from a first client device—to the contrary, the ’896 provisional suggests that such a transfer need only take place *manually* “when a customer browses or searches through his set of documents.” *Id.*, 17. To the extent the ’896 provisional discloses sending metadata at all, it discloses it in the context of MediaSpace’s directory update system ensuring that a customer may see a file that “ha[s] not yet been cached locally on the device,” which does not require use of metadata received from the client on which the file was modified. *Id.*

6. “determine that the server system is in communication with the first client device associated with the user”

88. Claim 5 of the ’607 patent recites this limitation. For the reasons discussed above in connection with the same limitation recited in the ’823 patent’s Challenged Claims, this limitation is not disclosed in the ’896 provisional.

7. “automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device”

89. Claim 5 of the ’607 patent recites this limitation. For the reasons discussed above in connection with the same limitation recited in the ’823 patent’s Challenged Claims, this limitation is not disclosed in the ’896 provisional.

8. **“wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device”**

90. Claim 7 of the '607 patent recites this limitation. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.

9. **“wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file”**

91. Claim 17 of the '607 patent recites this limitation. The '896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not disclose the use of this functionality in the context of the alleged “MediaSpace” system or invention. '896 provisional, 14. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). Neither the word “replace” (nor any of its other forms) nor the concept of replacing an older file with a newer one appears anywhere in the '896

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

provisional in the context of the allegedly inventive “MediaSpace” system. In addition, the ’896 provisional does not disclose receiving a copy of a file from one client device and using it to replace an older version of the same file on another client device. This limitation is not disclosed in the ’896 provisional.

D. The Challenged Claims of the ’622 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional

92. The Challenged Claims of the ’622 patent each recite one or more limitations that are not disclosed in the ’896 provisional.

1. **“receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device”**

93. Each Challenged Claim of the ’622 patent recites this limitation, in substantially identical form (claim 11 adds that the “receiving” is “by the server system,” claim 16 adds that the “receiving” is “by [a] first computer system,” and claim 16 also refers to a “second computer system” instead of a “first client device”—none of these alterations change the relevant analysis). This limitation is not disclosed in the ’896 provisional. The ’896 provisional says nothing about automatically receiving a file from a first client device responsive to the user modifying content of the file. The ’896 provisional discusses automated backups in the context of prior art systems, but these processes are not described as backing up

files in response to a user modifying content of any file. '896 provisional, 15. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

94. The '896 provisional also discusses prior art distributed file systems that could “automatically cach[e] often-used files on the local file system” and “only synchroniz[e] them when they have been changed.” '896 provisional, 14. However, this excerpt describes receiving a file **at a client device** that might be a copy of the file (i.e., cached) that is also stored remotely, not receiving a file **from** a client device in response to a user modifying the file. *Id.* (“Additionally, **accessing a file located on a distributed file system** tends to be slower than accessing files on the local hard drive. . . . [T]his separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if **the file he is remotely accessing** is the latest version of the file, versus an earlier one that has been marked to be updated.”).⁹ And, again, this discussion only appears in

⁹ For similar reasons, the limitation in claim 5 of the '622 patent reciting: “receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device

the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

2. **“receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file” and “automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device”**

95. Each Challenged Claim of the '622 patent recites these limitations, in substantially identical form (claim 11 adds that the “receiving” and “transferring” are “by the server system,” claim 16 adds that the “receiving” is “by [a] first computer system, from [a] second computer system,” and claim 16 also recites “transferring” by a “first computer system” “to a third computer system”—none of these alterations change the relevant analysis). The '896 provisional does not disclose receiving metadata from a client that has already been assigned a “first priority greater than a second priority” assigned to a copy of the file associated with

responsive to the user modifying a content of the second file stored on the second client device” is not disclosed in the '896 provisional.

the metadata. Although the '896 provisional discusses a “directory update system” that “updates at higher priority than the documents to be synchronized,” it discusses this directory update system in the context of updating MediaSpace’s “single logical area for customers to store and access their documents.” '896 provisional, 16-17. The '896 provisional does not describe the client as generating or transferring any metadata or assigning it any priority. In addition, the '896 provisional does not disclose ***automatic*** transfer of first metadata from a server to a “second client device” after the server receives the first metadata from a first client device—to the contrary, the '896 provisional suggests that such a transfer need only take place ***manually*** “when a customer browses or searches through his set of documents.” *Id.*, 17. To the extent the '896 provisional discloses sending metadata at all, it discloses it in the context of MediaSpace’s directory update system ensuring that a customer may see a file that “ha[s] not yet been cached locally on the device,” which does not require use of metadata received from the client on which the file was modified. Thus, these limitations are not disclosed in the '896 provisional. *Id.*

3. **“automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device”**

96. Each Challenged Claim of the '622 patent recites this limitation, in

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

substantially identical form (claim 11 adds that the “transferring” is “by the server system,” claim 16 adds that the “transferring” is “by [a] first computer system,” claim 16 refers to a “third computer system” as the recipient of the “copy of the first file,” and claim 16 refers to the “first file” being “from [a] second computer system”—none of these alterations change the relevant analysis). The ’896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not disclose the use of this functionality in the context of the alleged “MediaSpace” system or invention. ’896 provisional, 14. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). Neither the word “replace” (nor any of its other forms) nor the concept of replacing an older file with a newer one appears anywhere in the ’896 provisional in the context of the allegedly inventive “MediaSpace” system. In addition, the ’896 provisional does not disclose receiving a copy of a file from one client device and using it to replace an older version of the same file on another client device. Thus, this limitation is not disclosed in the ’896 provisional.

4. **“wherein availability of the version of the first file is presented at the second client device based on the first metadata”**

97. Claim 4 of the '622 patent recites this limitation. This limitation is not disclosed in the '896 provisional. The '896 provisional does not disclose receiving metadata from a client that has already been assigned a “first priority greater than a second priority” assigned to a copy of the file associated with the metadata. Although the '896 provisional discusses a “directory update system” that “updates at higher priority than the documents to be synchronized,” it discusses this directory update system in the context of updating MediaSpace’s “single logical area for customers to store and access their documents.” '896 provisional, 16-17. The '896 provisional does not disclose the presentation of availability of any version of any file based on metadata received from any client device. Instead, the presentation of availability of a version of a file in the MediaSpace system could be based on metadata generated at and sent from a MediaSpace server. In addition, the '896 provisional does not disclose **automatic** transfer of first metadata from a server to a “second client device” after the server receives the first metadata from a first client device—to the contrary, the '896 provisional suggests that such a transfer need only take place **manually** “when a customer browses or searches through his set of documents.” *Id.*, 17. To the extent the '896 provisional discloses sending metadata

at all, it discloses it in the context of MediaSpace's directory update system ensuring that a customer may see a file that "ha[s] not yet been cached locally on the device," which does not require use of metadata received from the client on which the file was modified. *Id.*

5. "determine that the server system is in communication with the first client device associated with the user"

98. Claim 5 of the '622 patent recites this limitation. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.

6. "automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device"

99. Claim 5 of the '622 patent recites this limitation. For the reasons discussed above in connection with the same limitation recited in the '823 patent's Challenged Claims, this limitation is not disclosed in the '896 provisional.

7. "wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device"

100. Claim 7 of the '622 patent recites this limitation. For the reasons discussed above in connection with the same limitation recited in the '823 patent's

Challenged Claims, this limitation is not disclosed in the '896 provisional.

8. **“initially determining, by the server system, that the server system is not in communication with the second client device associated with the user”**

101. Claim 12 of the '622 patent recites this limitation. This limitation is not disclosed in the '896 provisional. The '896 provisional does not disclose any determination that any server is or is not in communication with any particular client device. In fact, neither the word “communication” nor any of its related forms appears anywhere in the '896 provisional.

9. **“automatically transferring, by the server system, the copy of the first file to the second client device to replace the older version of the first file stored on the second client device with the copy of the first file, responsive to (i) resuming communication with the second client device, (ii) determining that the server system is in communication with the second client device, (iii) and receiving the copy of the first file from the first client device”**

102. Claim 12 of the '622 patent recites this limitation. This limitation is not disclosed in the '896 provisional. The '896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not disclose the use of this functionality in the context of the alleged “MediaSpace” system or invention and does not disclose synchronization that is responsive to the resumption of communication between a server and client. '896 provisional, 14. This discussion only appears in the context of criticism of the prior

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, at 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). The ’896 provisional also does not disclose any determination of whether any server system is in communication with any client device. Neither the word “replace” (nor any of its other forms) nor the concept of replacing an older file with a newer one appears anywhere in the ’896 provisional in the context of the allegedly inventive “MediaSpace” system. In addition, the ’896 provisional does not disclose receiving a copy of a file from one client device and using it to replace an older version of the same file on another client device.

E. The Challenged Claims of the ’942 Patent Each Recite One or More Limitations that are Not Disclosed in the ’896 Provisional

103. The Challenged Claims of the ’942 patent each recite one or more limitations that are not disclosed in the ’896 provisional.

1. **“receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file” / “receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file”**

104. Each Challenged Claim of the '942 patent recites these limitations, in substantially identical form. These limitations are not disclosed in the '896 provisional. The '896 provisional says nothing about automatically receiving a file from any electronic device responsive to a user modifying content of the file. The '896 provisional also does not disclose any application that sends or receives a file responsive to a user modifying content of the file. The '896 provisional discusses automated backups in the context of prior art systems, but these processes are not described as backing up files in response to a user modifying content of any file. '896 provisional, 15. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, at 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

105. The '896 provisional also discusses prior art distributed file systems that could “automatically cach[e] often-used files on the local file system” and “only synchroniz[e] them when they have been changed.” '896 provisional, 14. However, this excerpt describes receiving a file **at a client device** (e.g., the “second electronic device” or “third electronic device” recited in the Challenged Claims of the '942 patent) that might be a copy of the file (i.e., cached) that is also stored remotely, not receiving a file **from** a client device (“second electronic device” or “third electronic device”) in response to a user modifying the file. *Id.* (“Additionally, **accessing a file located on a distributed file system** tends to be slower than accessing files on the local hard drive. . . . [T]his separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if **the file he is remotely accessing** is the latest version of the file, versus an earlier one that has been marked to be updated.”). And, again, this discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

2. **“determine whether the first electronic device is in communication with a third electronic device” / “determine whether the first electronic device is in communication with the second electronic device”**

106. Each Challenged Claim of the '942 patent recites these limitations, in substantially identical form. These limitations are not disclosed in the '896 provisional. The '896 provisional does not disclose any determination that any device is in communication with any other device. In fact, neither the word “communication” nor any of its related forms appears anywhere in the '896 provisional.

3. **“automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device” / “automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device”**

107. Each Challenged Claim of the '942 patent recites these limitations, in substantially identical form. These limitations are not disclosed in the '896 provisional. The '896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not

disclose the use of this functionality in the context of the alleged “MediaSpace” system or invention and does not disclose the automatic forwarding of any modified copy of a file received from any “second” or “third” device by any “first device” to any respective “third” or “second” device. ’896 provisional, 14. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). The ’896 provisional also does not disclose any application that performs this functionality.

4. **“wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file” / “wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file”**

108. Each Challenged Claim of the ’942 patent recites these limitations, in substantially identical form. These limitations are not disclosed in the ’896

provisional. Neither the word “replace” (nor any of its other forms) nor the concept of replacing an older file with a newer one appears anywhere in the ’896 provisional in the context of the allegedly inventive “MediaSpace” system. In addition, the ’896 provisional does not disclose receiving a copy of a file from one device and using it to replace an older version of the same file on another device.

5. **“wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device”**

109. Claims 4 and 13 of the ’942 patent recite this limitation. This limitation is not disclosed in the ’896 provisional. The concept of replacing an older file with a newer one does not appear anywhere in the ’896 provisional in the context of the allegedly inventive “MediaSpace” system. In addition, the ’896 provisional does not disclose receiving a copy of a file from one device and using it to replace an older version of the same file on another device.

F. The Challenged Claims of the '561 Patent Each Recite One or More Limitations that are Not Disclosed in the '896 Provisional

110. The Challenged Claims of the '561 patent each recite one or more limitations that are not disclosed in the '896 provisional.

- 1. “a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user”**

111. This limitation appears in claim 1 of the '561 patent and is not disclosed in the '896 provisional. The '896 provisional does not disclose “second” and “third” electronic devices being associated with a user and does not disclose a “first electronic device being in communication” with such devices. In fact, neither the word “communication” nor any of its related forms appears anywhere in the '896 provisional.

2. **“receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file” / “receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file” / “receiving from the first electronic device a copy of a first electronic file modified by the user, the copy of the modified first electronic file being automatically provided to the electronic system by the first electronic device when a content of the first electronic file is modified by the user” / “receiving from the second electronic device a copy of a second electronic file modified by the user, the copy of the modified second electronic file being automatically provided to the electronic system by the second electronic device when a content of the second electronic file is modified by the user”**

112. These limitations appear in claims 1 and 8 of the '561 patent and are not disclosed in the '896 provisional. The '896 provisional says nothing about using any application to automatically transfer any file from a client device (“first electronic device” (in claim 8 only), “second electronic device,” or “third electronic device” (in claim 1 only)) responsive to a user modifying content of the file on the client device. The '896 provisional discusses automated backups in the context of prior art systems, but these processes are not described as backing up files in response to a user modifying content of any file. '896 provisional, 15. This discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art

solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

113. The ’896 provisional also discusses prior art distributed file systems that could “automatically cach[e] often-used files on the local file system” and “only synchroniz[e] them when they have been changed.” ’896 provisional, 14. However, this excerpt describes receiving a file **at a client device** (e.g., “first electronic device” (in claim 8 only), “second electronic device” or “third electronic device” (in claim 1 only))) that might be a copy of the file (i.e., cached) that is also stored remotely, not receiving a file **from** a client device (“first electronic device” (in claim 8 only), “second electronic device,” or “third electronic device” (in claim 1 only))) in response to a user modifying the file. *Id.* (“Additionally, **accessing a file located on a distributed file system** tends to be slower than accessing files on the local hard drive. . . . [T]his separate synchronization step introduces another problem: because the synchronization process can be lengthy, the customer is never entirely sure if **the file he is remotely accessing** is the latest version of the file, versus an earlier one that has been marked to be updated.”). And, again, this discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or

another [are] inadequate to the task”); *id.*, at 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

3. **“automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user” / “automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user” / “automatically transferring the modified first electronic file copy to the second electronic device to replace an older version of the first electronic file stored on the second electronic device with the modified first electronic file copy having the content modified by the user” / “automatically transferring the modified second electronic file copy to the first electronic device to replace an older version of the second electronic file stored on the first electronic device with the modified second electronic file copy having the content modified by the user”**

114. These limitations appear in claims 1 and 8 of the '561 patent and are not disclosed in the '896 provisional. The '896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not disclose the use of this functionality in the context of the alleged “MediaSpace” system or invention and does not disclose the automatic transferring of a file after modification by a user. '896 provisional, 14. This discussion only appears in the context of criticism of the prior art, not in connection with any

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). Neither the word “replace” (nor any of its other forms) nor the concept of replacing an older file with a newer one appears anywhere in the ’896 provisional in the context of the allegedly inventive “MediaSpace” system. In addition, the ’896 provisional does not disclose receiving a copy of a file from one client device and using it to replace an older version of the same file on another client device.

4. **“wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file” / “wherein the first electronic device automatically transfers the copy of the modified first electronic file to the electronic system upon determining that a save operation has been performed on the modified first electronic file”**

115. These limitations appear in claims 1 and 8 of the ’561 patent and are not disclosed in the ’896 provisional. The ’896 provisional discusses synchronizing files “when they have been changed” in the context of prior art distributed file systems but does not disclose the use of this functionality in the context of the alleged “MediaSpace” system or invention and does not disclose the automatic transferring of a file “upon determining that a save operation has been performed.” ’896 provisional, 14. This discussion only appears in the context of criticism of the prior

art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, at 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art). The only reference to saving files in the ’896 provisional appears in the context of a discussion of prior art email systems and has nothing to do with this limitation. *Id.*, at 15 (“If the customer, for example, were to open the file, change it, then save it back out, the results would be ambiguous to the user because the email application, not the user, specified its location. Usually, the saved file would end up buried in an email file cache in an undisclosed area of the file system.”). So again, this discussion only appears in the context of criticism of the prior art, not in connection with any description of the alleged invention. *Id.*, 13-16 (describing prior art solutions that “in one way or another [are] inadequate to the task”); *id.*, at 16-19 (describing how the allegedly inventive “MediaSpace” product addresses such deficiencies in the art).

5. **“wherein the first application is configured to store the modified first electronic file copy to the memory device associated with the first electronic device when the third electronic device is not in communication with the first electronic device” / “wherein the electronic system is configured to store the modified first electronic file copy to the storage component when the second electronic device is not in communication with the electronic system”**

116. These limitations appear in claims 4 and 11 of the ’561 patent,

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

respectively, and are not disclosed in the '896 provisional. The '896 provisional does not disclose any determination of whether any device is or is not “in communication” with any other device.

6. **“the electronic system being coupled to a first electronic device having stored thereon a first electronic file and a second electronic device having stored thereon a second electronic file, the first electronic device and the second electronic device each being associated with a user”**

117. This limitation appears in claim 8 of the '561 patent and is not disclosed in the '896 provisional. The '896 provisional does not disclose difference devices of a user that are “coupled to” a “first electronic device” as described in this limitation.

VI. THE CHALLENGED CLAIMS OF THE '823, '787, '607, '622, '942, AND '561 PATENTS WOULD HAVE BEEN OBVIOUS IN VIEW OF SIGURDSSON IN COMBINATION WITH ONE OR MORE OF MOROMISATO, RAO AND/OR SHAPPELL

A. Summary of the Prior Art

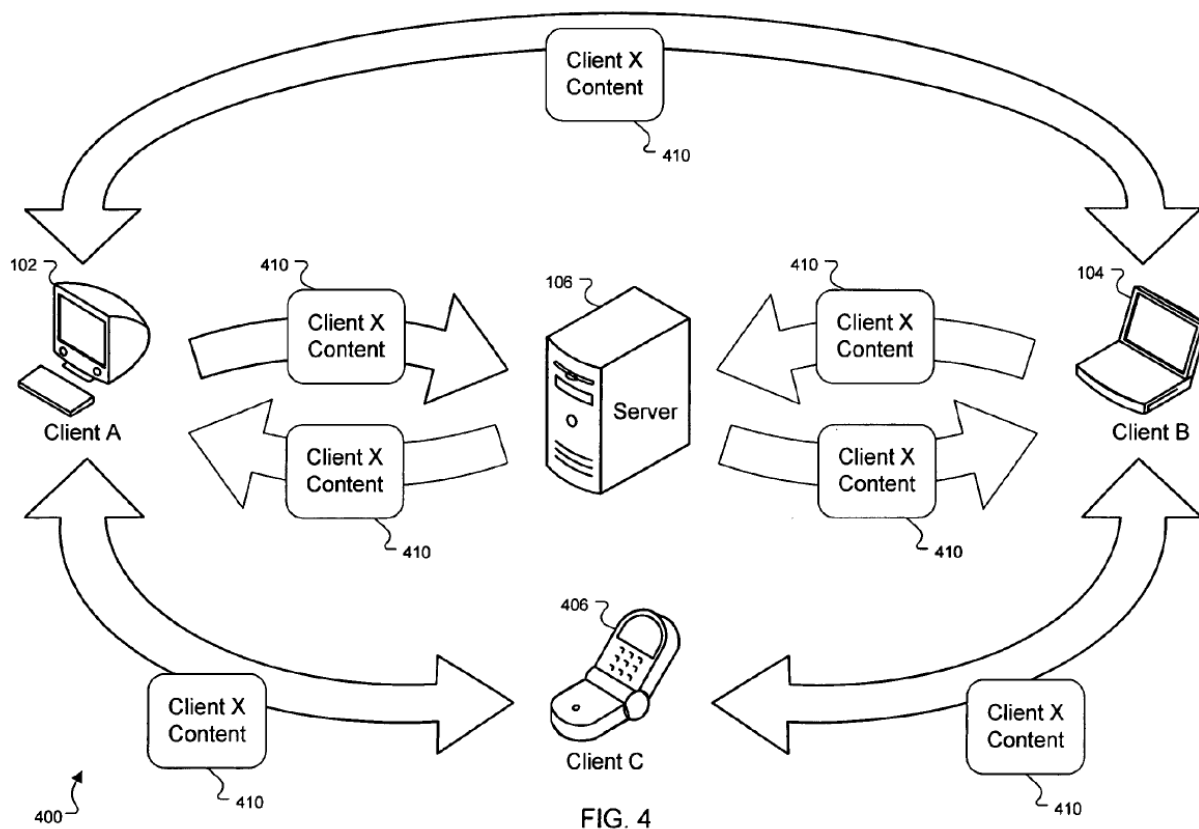
1. Sigurdsson [EX1016]

118. Sigurdsson is a U.S. patent application publication that was assigned to Google. See EX1017, Sigurdsson File History, 10/7/2021 Notice Regarding Change of Power of Attorney (“Under 37 CFR § 3.73(b), Google LLC, a Delaware limited liability company, certifies that [it is] the assignee of 100% of the right, title, and interest in the patent applications listed on the attached Appendix (Table 1) by virtue

of assignments, including assignments from inventors of the patent applications”
(listing Sigurdsson among the assigned applications).).

119. Sigurdsson discloses systems and methods for synchronizing files across multiple client devices of a user by way of a server. Sigurdsson, ¶¶ 26-27. For example, as shown in Figure 1, “Client A 102 and the Client B 104 represent computers which a user may employ to manage digital information, for example by saving or opening files, or accessing web pages.” *Id.* ¶ 27. “A user may access many computers including, for example, computers at home, computers at work, and mobile computers, such as personal digital assistants (PDAs) and cell phones.” *Id.* Sigurdsson explains that “Server 106” in Figure 1 “can provide a temporary storage location, or ‘drop box’, for a user’s files, documents, web pages, or the contents of these documents.” *Id.* “The drop-box can facilitate sharing this information between the user’s computers by storing the shared content if the target computer is offline.” *Id.* “Later, when the target computer comes online, the Server 106 may transmit the content to the target computer.” *Id.*

120. One embodiment of Sigurdsson’s invention appears in Figure 4:



“Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106,” Client C can “communicate with the Server 106 in a manner similar to the method used by the Clients A or B.” (Sigurdsson, ¶ 51.) Sigurdsson explains that Clients A, B, and C may be associated with the same user. (*Id.* ¶ 58.) Each of Sigurdsson’s clients may be remote from the server and “typically interact” with the server “through a network,” such as the internet. (*Id.* ¶¶ 69, 112-13.)

121. In one embodiment, Sigurdsson explains that “Client B” may be a user’s “work computer” and that, when a user changes a file on the user’s work

computer, the file may be automatically sent to “Server 106,” which, in turn, may send the updated file to the user’s home computer (e.g., “Client A”). Sigurdsson, ¶ 28; *see also id.* ¶ 45 (“The Client A 102 posts the Client A Content 108 to the Server 106 after content is changed on the Client A 102.”). In this example, “Server 106 serves as a conduit for sharing file content across the user’s work and home computers” and may “facilitate synchronization of content among the user’s computers.” *Id.* ¶¶ 28, 31.

122. Sigurdsson describes an “event listener” that can detect events, such as saving or editing of a file by a user. Sigurdsson, ¶ 32; *see also id.* ¶ 45 (“The Client A 102 posts the Client A Content 108 to the Server 106 after content is changed on the Client A 102.”); *id.* ¶ 47 (“The Event Listener 322 listens for and detects user actions, such as saving a file, deleting a file, or viewing a web page, that may require the associated content Client A Content 108 to be posted to the Server 106.”). After detecting such an event, content related to the file is uploaded to the server. *Id.* ¶ 34. A different client may then poll the server for new content, including the new content resulting from the file save or edit. *Id.* ¶ 37. Sigurdsson explains that the “client” systems it describes are fungible and explains that, for any particular user, the “synchronization steps” described in Sigurdsson may be used by “any of the user’s clients.” *Id.* ¶ 44; *see also id.* ¶ 50 (“Both Clients A and B 102, 104 can contain

similar components and can accept and transmit content in a similar manner. For example, both clients can include Files 346, HTML Files 348, an Index 350, an Event Listener 352, a File to Text/HTML Converter 354 and an HTTPS Client/server 356.”). For example, Sigurdsson’s clients may “each represent a computer used by a user including, for example, a personal computer at home, a [sic] computers at work, and portable computers, such as a laptop computer, a PDA, and a cell phone.” *Id.* ¶ 45.

123. The “Server 106” described in Sigurdsson may include applications, such as a “Synchronization Manager 332” application. Sigurdsson, ¶ 48.

124. The content sent from clients to Server 106 in Sigurdsson may include entire copies of files and metadata. Sigurdsson, ¶ 54 (“In step 506, the content originating on the client is posted to the server. The content can include a copy of the document (or the content of the document) that was saved by the user as well as metadata that describes information, such as the type of data, its source, when it was saved, and when it was posted to the server.”).

125. Sigurdsson explains that different types of files may be prioritized differently for synchronization. Sigurdsson, ¶ 58. For example, Sigurdsson explains that smaller files may be prioritized over larger files such that smaller files are prioritized before larger files. *Id.* Newer files could be assigned higher priorities

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

for synchronization than older files. *Id.* And certain types of files could be prioritized over other types of files. *Id.*

126. Sigurdsson explains that, in certain embodiments, metadata may be sent to Server 106 instead of actual files, particularly when the files are large. Sigurdsson, ¶ 72. Server 106 can then send a list of missing content, including the metadata, to a client that has been offline when such metadata was sent. *Id.* Sigurdsson explains that clients may push new data as soon as it is generated. *Id.* ¶ 84.

2. Moromisato [EX1018]

127. Moromisato discloses a system for synchronizing “workspace” folders in which a client may communicate file changes to a server, which, when updated, automatically sends a command to all other clients to synchronize their data with the latest version: “in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. The server modifies its copy of the data and then transmits the command to all collaborators. Each collaborator uses the command to update its local data copy.” Moromisato, 1:47-54. And, “[i]n a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based

collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.” *Id.*, 14:47-55.

128. More particularly, Moromisato teaches the automatic transfer of a modified first electronic file copy to an electronic device to replace an older version of the first electronic file stored on the electronic device with the modified first electronic file copy having the content modified by the user. For example, Moromisato teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system “sites”. The automatic synchronization mode requires that changes to a file be synchronized as they happen. . . . Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. If a file has been edited, then the document share engine may either disseminate the entire updated file contents or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

129. Moromisato was assigned to Microsoft (see the first page of Moromisato, item (73)) and describes functionality related to the then- and now-popular Microsoft Windows operating system. For example, Moromisato discloses “synchronizer” functionality designed to operating in conjunction with the Microsoft Windows operating system. Moromisato, 7:9-24 (“The file system RAMP program 304 is part of the collaborative system code and detects changes to files in the OS file system 302. It communicates with the file synchronizer 308, as shown schematically by arrow 340, in order to inform the file synchronizer 308 that it needs to synchronize those changes. In particular, the Windows XP Pro operating system provides a system event notification system to which clients can subscribe to receive notifications of selected events. These events include changes in the OS file system, such as folder and file edits, additions and deletions. . . . During the collaborative system start up, the file system RAMP program 304 is invoked so that it can register with the OS event notification system for file notifications indicating changes to files and file contents in the Windows shell file system 302.”).

130. One embodiment of Moromisato’s file synchronizer is illustrated in Figure 3:

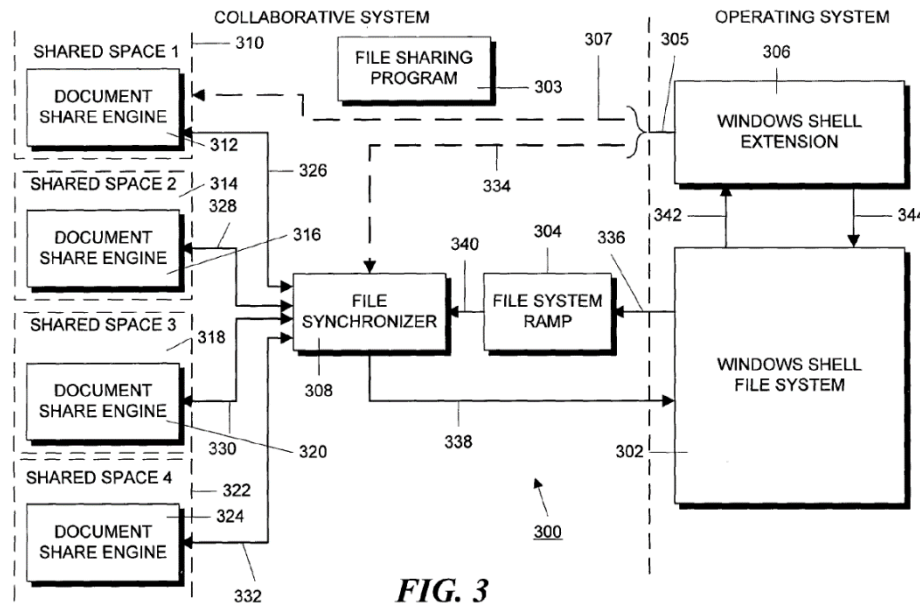


FIG. 3

Id., Fig. 3; see also Moromisato, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system 302 and vice versa.”).

131. Moromisato discloses using an application (i.e., the “file synchronizer 308” shown in Figure 3) that is configured to automatically transfer a modified electronic file copy to another electronic device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44 (“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer replaces an older version of the first electronic file stored on the third

electronic device with a modified first electronic file copy having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents”); *see also id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing).

132. The shared spaces in Moromisato each have an associated document share engine 312, 316, 320 and 324, respectively. Each document share engine stores a “file descriptor” for each file in a synchronized folder. “The file descriptor consists of the file metadata (such as filename, size, and modification time) and is used to keep track of the file in the collaborative system. During collaborative system operation, each engine also sends data change requests containing the file descriptors to each collaborator in its respective shared space in order to keep all collaborators synchronized.” Moromisato, 8:1-8.

133. Moromisato discloses “stub” files that represent files available from another computer but that have not yet been downloaded onto the user’s device. Moromisato, 14:47-56. Moromisato explains that download settings on the user’s device may determine whether to: “(1) Always download new files; (2) Download

new files if the file sizes are smaller than a selected number of kilobytes (where the selected number of kilobytes can be entered into textbox 944) and (3) Download new files manually as specified by the user.” *Id.*, 15:6-10. “If the download settings are such that file downloading is deferred to a later time, that file is represented in the OS file system as a ‘stub’ file that is very similar to a Windows shortcut (.Ink file) in that it represents a target file with a small file that contains information necessary to find the target file. The stub file is represented in the user interface by the same icon as the target file, but it may have an overlay of some sort that indicates that it is a stub file in a manner similar to the hand overlay shown in FIG. 1.” *Id.*, 15:14-22. As explained in Moromisato, the stub file on a first client is created whenever a second client modifies a shared file that was previously stored on the first client.

134. Moromisato discloses “stub” files that represent files available from another computer but that have not yet been downloaded onto the user’s device. Moromisato, 14:47-56 (“In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded***

to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.”).

135. Moromisato’s representations of stub files may be replaced with representations of real files. Moromisato, 15:36-38 (“In one embodiment, ***the user can download the target file associated with a stub file by double clicking on the stub file icon.***”), *id.*, 16:27-51 (“[T]he user/invitee at some point will download the file either by double clicking on it or by selecting the file in a list and then using the file tasks pane to download the file. The Windows shell extension 306 is invoked as discussed above and accesses the collaborative system via the web services interface. It uses the files tool in the shared space to fetch the file. When a file is being downloaded, a download pane 1370 such as that illustrated in FIG. 13 appears in the synchronize pane 1320. The download pane 1370 displays the status of the download, for example, by means of a progress bar 1372 or other means. A “Cancel” button 1374 allows the user to cancel the download. The download pane 1370 always applies to the selected file. If there is no selection but some files are being downloaded then, in one embodiment, the download pane may show a statement such as “Downloading 3 files” in place of the progress bar 1372. If there are multiple files selected and all the files are stubs being downloaded then a similar statement can be shown. The information in this pane is provided by the Windows shell

extension that, in turn, uses the files tool in the shared space (via the web services interface) to get the required information. When the file arrives, the stub file representing the actual file is replaced by the actual file and the file system snapshot is updated to indicate that the file contents have changed.”).

3. Rao [EX1019]

136. Rao discloses a system for synchronizing files across multiple user devices including a server that receives, from a user device, metadata associated with a version of a file that is generated from the user modifying the content of the file, the metadata being assigned a first priority greater than a second priority assigned to a copy of the file. For example, Rao discloses synchronizing files between a client and a server using a two-step process: first synchronizing metadata associated with files and then synchronizing the “raw data” of the actual files themselves. Rao, ¶ 63 (“In the first phase, metadata is synchronized between a server and client while the raw data remains on the server system. The client system, when connected or disconnected, may use and manipulate the metadata for any purpose whatsoever, including selecting one or more raw data files that are to be available on the client system. The second phase of synchronization is to synchronize the selected raw data to the client system. The synchronized raw data may be manipulated, changed, and used on the client system when disconnected. Upon reconnect, any changes made

to the metadata or the raw data are synchronized between the systems.”).

137. Thus, Rao discloses a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files themselves, such that a server in one of Rao’s embodiments may assign a first priority to metadata and a second priority to a copy of a file associated with such metadata based on the priority assignment configuration.

138. Rao explains that transmitting metadata first, before the raw data, is useful because it allows certain functions to be performed before the raw data is present. Rao, ¶ 43 (“Because the metadata relating to file 140 is contained in file backed item 132, certain functions and operations may be available to the user of the client system 104. Using the example of an audio file contained in file 140, the client system 104 may be able to display the various audio tracks, rate the individual tracks, group the tracks into playlists or other useful groupings, and perform other operations using only the metadata contained in item 132.”). In one embodiment, Rao’s metadata may be used to indicate whether a file’s raw data should be synchronized, which involves prioritizing the transfer and synchronization of metadata first, given that an indication for synchronizing raw data wouldn’t be available without the metadata. *Id.* ¶ 44 (“[M]etadata contained in item 138 may be used to manipulate various aspects of file 140. For example, the item 132 may be

used to indicate that file 140 is to be synchronized rather than ghosted.”). Rao further discloses synchronization takes place automatically when a client and server are reconnected. *Id.* ¶ 62 (“When the client and server are reconnected in block 402, any changes to the items [i.e., metadata] are synchronized[.]”).

139. Rao discloses that metadata may be transferred to a client and used to display information about a file that is available on a server but has not yet been downloaded to the client. *See, e.g.*, Rao, ¶ 43. For such files, an icon may be displayed to indicate that the file is “ghosted.” *Id.* ¶ 66 (“While the systems are disconnected in state 325, placeholders may be used for the files 322 and 324. The files are ‘ghosted’ and may or may not be seen in the directory structure. For example, ghosted files may be displayed as grayed out or semi-transparent.”). Rao discloses that any “ghosted files” that are flagged to be synchronized may be downloaded from a server during the next synchronization. Rao, ¶ 67 (“When the connection is restored in state 326, any changes to the item repositories 310 and 318 are synchronized, and then any ghosted files that are flagged, such as file 324, are synchronized. The fully synchronized file 324 is then transferred to the client 304 and usable when the client 304 is disconnected from the server 302.”). Rao explains that a user may indicate that they want raw data associated with a file to be delivered to the user’s device on the next synchronization. *Id.* ¶ 65 (“The user may

additionally flag file 324 to be synchronized.”).

140. As discussed in Rao, metadata files are used in place of raw data because “the raw data may be extraordinarily large files which may require a long time to transfer between the server system 102 and the client system 104.” Rao, ¶ 48; *see also id.* ¶ 46 (“One of the practical benefits of ghosting files such as file 140 is that large databases having many raw data files may be much larger than the client storage 128 can hold. However, much of the usefulness of the data is still present because of the metadata contained in the various items stored in the item repository 130.”). A POSA would understand from Rao that metadata files are generally smaller and should be sent prior to potentially “large” raw data files.

4. Shappell [EX1020]

141. Shappell discloses “shared spaces” in which files may be shared among group members. Shappell, ¶ 76 (“For sharing files among group members, metadata is transmitted to all members when files become available, such as is shown in the steps 1904 and 1906 in the flow chart in FIG. 19. Such metadata are sufficient to render an icon of the shared file and may include date, time, thumb-nail information, name size, and optionally the source of the information.”). Shappell discloses that metadata of this type is sent first (with a higher priority) than the raw data associated with files to facilitate user-interface display of the availability of such files on a

user's device before such files are obtained and stored on the user's device. *Id.* ¶ 78 (“As shown, the name and path to the file are included in the metadata of a transmitted shared file. In addition, an identifier for the container of the shared file is provided as well as the location of the machine on which the file resides. An indication of the type of file, the file size and modification date are also included. Optionally, the transmitted file may include a thumbnail and Uniform Resource Locator for the icon associated with the shared file.”); *id.* ¶ 79 (“As is shown in the flow chart in FIG. 20, the system renders such a display upon receipt that metadata concerning a shared file is available in the peer network, as is shown in method step 2002. Other files that are ‘missing’ may be unavailable to the user, such as when a system that is hosting the file of interest is offline. In this instance, a display icon 1506 is presented as a ghosted icon that includes a small image of a red ‘X’ or other suitable visual indicia that the file is unavailable to the user.”). Only after the metadata is sent is the raw data of the file sent in this embodiment from Shappell. *Id.* ¶ 82 (“In order for a group space member to access an updated [file] that is not locally stored, the file must first be transmitted to the local machine. FIG. 16 illustrates a graphical interface of a file copy dialog 1600 presented to the user for indicating that a transfer operation is occurring. When the file is available on the local machine, the icon providing a shortcut to the shared file will be updated to

reflect the file transfer.”). Shappell discloses that such synchronization takes place automatically. *Id.* ¶ 76(“For sharing files among group members, metadata is transmitted to all members when files become available[.]”).

142. Shappell discloses, with respect to its “group folder” functionality: “[B]ecause content in a group folder is likely to change over time and the organization of content does not enable the user to discern what content is new or has been recently altered, a ‘new’ glyph may be presented on folders and files that are new or have been recently changed. This glyph provides a visual indicator of new content.” Shappell, ¶ 80. Shappell further discloses that special icons are shown to “inform a user of a group shared space file that is not present locally.” *Id.* ¶ 79. “FIG. 15 illustrates a graphical interface 1500 for a Shared Space folder that contains files that are not local. As shown, a left hand pane 1502 presents a display icon 1504 to denote a ‘missing’ file that is available on another system. This icon is ghosted or semi-transparent. As is shown in the flow chart in FIG. 20, the system renders such a display upon receipt that metadata concerning a shared file is available in the peer network, as is shown in method step 2002. Other files that are ‘missing’ may be unavailable to the user, such as when a system that is hosting the file of interest is offline. In this instance, a display icon 1506 is presented as a ghosted icon that includes a small image of a red ‘X’ or other suitable visual indicia

that the file is unavailable to the user.” *Id.* ¶ 79.

143. Shappell further discloses that such a graphical indication could be made “upon receipt” of “metadata concerning a shared file” that “is available.” Shappell, ¶ 81; *see also id.* (“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will remove the local copy of the out-of-date file. This will change the visual representation of the file to a ‘missing’ file within the shared space folder when the application setting does not automatically update files, as is shown in step 2010 in FIG. 20. On the other hand, if automatic replication of the shared file is enabled, the local machine obtains an updated version of the file, as is shown in step 2006 in FIG. 20. Finally, an appropriate visual indicator is presented to the user as shown in step 2008.”). This functionality is illustrated in Figure 20 of Shappell.

144. Shappell discloses an automatic replication option that, when enabled, causes an old version of a shared file stored locally on a user device to be deleted and replaced with a new version of the file when the shared file is modified on another user device. Shappell, ¶ 81 (“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps

1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will remove the local copy of the out-of-date file.”).

B. '823 Ground 1: Claims 1-4, 8-11, and 13-16 of the '823 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Rao.

1. Independent Claim 1

145. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1d] automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[1e] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and

[1f] (ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and

[1g] wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

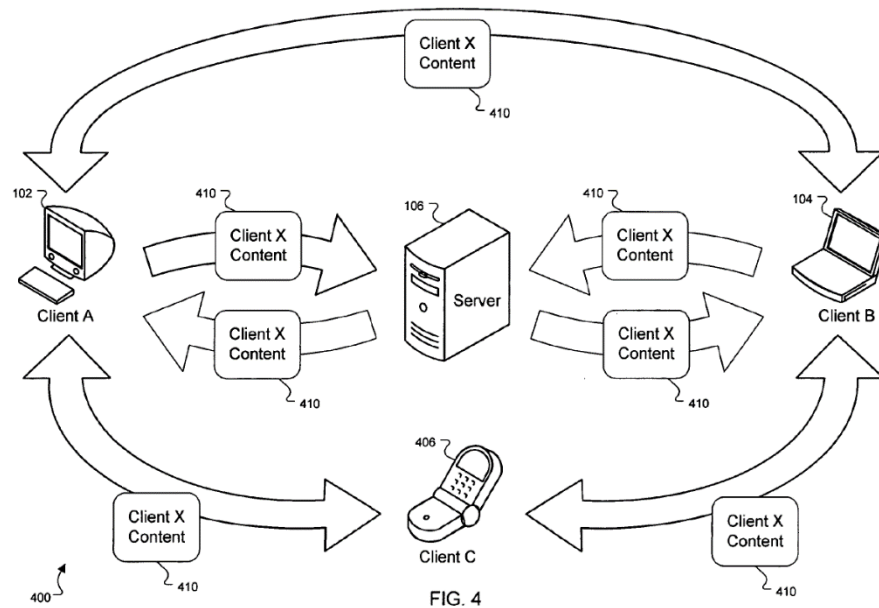
[1h] subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,

[1i] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

b. [1p]: “A system comprising:”

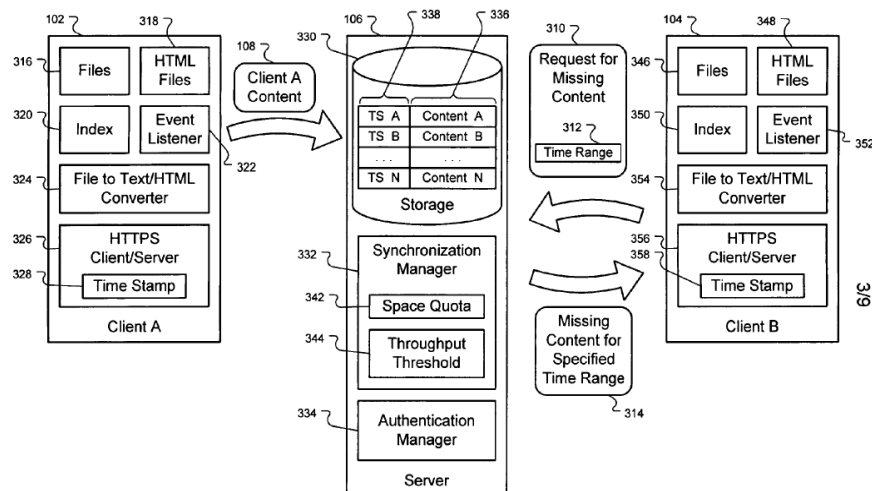
146. The preamble of claim 1 recites “[a] system.” I understand that the preamble of a claim generally does not limit the claim, but to the extent that the preamble provides a claim limitation, it is disclosed by Sigurdsson.

147. Sigurdsson discloses a system for sharing and synchronizing content. For example, Sigurdsson discloses a system (“400”) in Figure 4:



148. “FIG. 4 is a schematic showing a **system 400** for sharing and synchronizing content using a . . . client/server architecture. In the **system 400**, content still can be shared among the Client A 102, the Client B 104 and a Client C 406 using the Server 106 in a method similar to that of method 200 of FIG. 2.” Sigurdsson, ¶ 51. Subcomponents contained inside Client A 102, Client B 104, and Server 106 are further detailed in Figure 3 and associated text:

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622



See also Sigurdsson, ¶¶ 45-50.

149. For these reasons and the reasons that follow, Sigurdsson renders obvious the claimed system.

- c. [1a]: “a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:”

150. Sigurdsson discloses a server system comprising one or more processors programmed with computer program instructions that can be executed.

151. For example, “Server 106” in Figure 4 (and as described in connection with Figure 9 and elsewhere in Sigurdsson) is a “server system.” For example, Sigurdsson discloses that “System 900” depicted in Figure 9 “may be included in . . . Server 106.” Sigurdsson, ¶ 104. System 900 includes conventional computing components and operation – “a **Processor 910**, a Memory 920, a Storage Device

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

930, and Input/Output Devices 940” and “[t]he **Processor 910 is capable of processing instructions for execution within the System 900.**” *Id.* ¶ 105. “The Processor 910 is capable of **processing instructions stored in the Memory 920** or on the Storage Device 930 to display graphical information for a user interface on the Input/Output Devices 940.” *Id.* Sigurdsson further discloses, in connection with System 900:

The apparatus can be implemented in a **computer program** product tangibly embodied in an information carrier, e.g., in a machine-readable storage device or in a propagated signal, **for execution by a programmable processor**; and **method steps can be performed by a programmable processor executing a program of instructions** to perform functions of the described implementations by operating on input data and generating output. The described features can be implemented advantageously in one or more computer programs that are executable on a programmable system including at least one programmable processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. **A computer program is a set of instructions that can be used, directly or indirectly, in a computer to perform a certain activity or bring about a certain result.**

Id. ¶ 109; see also *id.* ¶ 110 (“Generally, **a processor will receive instructions** and data from a read-only memory or a random access memory or both. The essential elements of a computer are **a processor for executing instructions** and one or more memories for storing instructions and data.”).

152. Sigurdsson further discloses that Server 106 includes a “Synchronization Manager 332.” Consistent with conventional computing

techniques, a POSA would have understood and found it obvious that “Synchronization Manager 332” would comprise one or more computer program instructions that, when executed by one or more processors in Server 106, would cause Server 106 to perform operations. Sigurdsson,. ¶ 48 (describing operations performed by Synchronization Manager 332); *see also id.*, ¶ 109 (*e.g.*, “The features described can be implemented in digital electronic circuitry, or in computer hardware, firmware, software, or in combinations of them.... [M]ethod steps can be performed by a programmable processor executing a program of instructions to perform functions of the described implementations....”). Sigurdsson explains that Server 106 “can be implemented in a computer system that includes a back-end component, such as a data server, or that includes a middleware component, such as an application server or an Internet server.” *Id.* ¶ 112. A POSA would have understood and found it obvious that such a server computer system would include one or more processors programmed with computer program instructions that can be executed. *See id.*, ¶¶ 109-110. A POSA would have further understood and found it obvious that the instructions, when executed, would cause the server system to perform the functionality recited in the claim, as explained below.

- d. [1b]: “receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file”

153. To start, Sigurdsson discloses a server system that “**receive[s], over a network, a copy of a first file from a first client device.**” Sigurdsson discloses a collection of **client devices** connected to the server system **over a network**. Sigurdsson, Fig. 1, ¶¶ 26-27 (“The system **100** includes a Client A **102**, a Client B **104**, and a Server **106**.”), 69 (“network that links the server and other clients, such as the Internet”) 112-13 (“The components of the system can be connected by any form or medium of digital data communication such as a **communication network**. Examples of communication networks include, e.g., a LAN, a WAN, and the computers and networks forming the Internet. The computer system can include clients and servers. A client and server are generally remote from each other and typically interact through a network, such as the described one.”).

154. Sigurdsson further teaches that its server system – which is connected to client devices over a network – can **receive a copy of a first file** stored at one of the client devices, such as Client A. Sigurdsson, ¶¶ 28 (“The Client A Content 108, *which can include a copy of the user’s file*, is sent automatically to the Server

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

106.”), 54 (“In step 506, the content originating on the client is posted to the server.

The content can include a copy of the document (or the content of the document)

that was saved by the user[.]”), 65 (discussing copy of file saved by user), 85 (same),

103; *see also id.* ¶ 8 (“[E]xtracting the content of the first file may include generating

a copy of the first file that retains the first file’s original file formatting.”). I note

that although some embodiments refer to peer-to-peer sending of files (e.g., from

Client A to Client B), Sigurdsson describes that files may be sent directly to the

server system by a client, including in the embodiment described in Figures 4-7. For

example, Client A may “***send[] the data directly to the server***, as shown by the

dashed line pointing from the step 606 to the step 610. For example, referring to

FIG. 4, the ***Client X Content 410 can be sent directly to the Server 106***” *Id.*

¶ 68; *see also id.* ¶¶ 84-85 (describing that Client A posts content to the Server).

155. A POSA would have appreciated and recognized Sigurdsson’s

teachings of the obvious advantages of a client/server arrangement. The server can

stay online while user computers periodically go offline, so that Server **106**

beneficially provides a “drop-box” for “a user’s files” to “***facilitate sharing this***

information between the user’s computers by storing the shared content if the

target computer is offline.” Sigurdsson, ¶¶ 27-29, 61 (server tracks clients’

online/offline status). A server may also have more available bandwidth relative to

clients for transferring data. *Id.*, ¶ 79. Sigurdsson makes clear the peer-to-peer and client/server pathways between the user's multiple client devices are interchangeable, rendering it obvious to synchronize through the server. *Id.*, ¶ 51 ("Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106," Client C can "communicate with the Server 106 in a manner similar to the method used by the Clients A or B."). For example, Sigurdsson explains that each client may include an "HTTPS Client/server" that may interchangeably be used for synchronization with Server 106 (*id.* ¶ 54 ("HTTPS Client/server 326 of the Client A 102 can transmit an HTTP post command containing the Client X Content 410 to the Server 106.")) *or* synchronization with other clients (*id.* ¶ 47 ("The HTTPS Client/server can also be used as a server to provide content to peer clients, which is described in more detail below.")). Even when clients can communicate peer-to-peer, Sigurdsson teaches that they can still also use Server 106 for content sharing. Sigurdsson, ¶¶ 51, 79 ("Direct content sharing among peer clients can make use of available resource capacities at the clients, thus reducing the bandwidth traffic at the Server 106. ***The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from any the*** [sic] ***Clients 102, 104 or 406.***").

156. Sigurdsson further teaches that a copy of the first file "**is automatically**

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

received from the first client device responsive to the user modifying a content of the first file stored on the first client device.” Sigurdsson explains, for example, that “[t]he Client A 102 posts Client A Content 108 to the Server 106 ***when content is changed*** on the Client A 102.” Sigurdsson, ¶ 84; *see also id.*, ¶ 28 (“The Client A Content 108, which can include a copy of the user’s file, is **sent automatically** to the Server 106.”). As one way to implement this feature, Sigurdsson describes that event listeners running on a client device listen for events such as when a user modifies a file, and in response to such events, the client device will **automatically** send a copy of the modified file to the Server. *Id.*, ¶¶ 63 (“***Processing can start in step 602 when an event occurs. . . . For example, referring to FIG. 4, a user at the Client A 102 may edit and save a document on his work computer.***”) 84-85 (“***The Client A 102 posts Client A Content 108 to the Server 106 when content is changed on the Client A 102. . . . [P]osting can occur as soon as an event is generated by an action*** For example, a client can maintain an open connection with a server and post as soon as an event is generated. . . The Client A Content 108 is received by and stored in the Server 106.”), 28 (“The Client A Content 108, which can include a copy of the user’s file, is **sent automatically** to the Server 106.”), 32-33 (“For example, . . . a user at the Client A 102 may edit and save his resume on his home computer. This action may generate an event that is detected by an event listener. . . .

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

For example, after the event triggered by an action is detected, the content from the file associated with the action can be extracted. The extracted content can be the text or images included in the file that was saved, accessed or ***modified.***”), 45 (“The Client A 102 posts the Client A Content 108 to the Server 106 ***after content is changed*** on the Client A 102.”), 47 (“The ***Event Listener 322 listens for and detects user actions***, such as saving a file, deleting a file, or viewing a web page, that may require the associated . . . Client A Content 108 to be posted to the Server 106.”).

157. Sigurdsson further teaches that the copy of the first file is “***an updated version of the first file that is generated from the user modifying the content of the first file.***” Sigurdsson explains, for example, that a user may edit or otherwise modify a file’s content to generate an updated version of the file, which is then transmitted to the server system when the user saves the modified file:

For example, ***a user may edit a Word document file at his home computer.*** When the user saves the document, the document’s content is automatically extracted and transmitted to the user’s computer at work. ***When the user searches for the modified document later at his work computer,*** the content is presented to the user even though the Word document was not previously stored on the user’s work computer.

Sigurdsson, ¶ 26; *see also id.*, ¶¶ 28 (“For example, the user may save a text file on his home computer. The Client A Content 108, which can include a copy of the user’s file, is sent automatically to the Server 106.”), 63-64 (“[A]n event is generated in response to an action performed by a user, such as, saving a file or document. . . .

For example, referring to FIG. 4, a user at the Client A 102 may edit and save a document on his work computer. . . . [T]he index is updated to reflect the new saved version of the user's document.”), 32 (“For example, referring to FIG. 1, a user at the Client A 102 may edit and save his resume on his home computer. This action may generate an event that is detected by the event listener.”). A POSA would have understood that performing a file-save operation would result in modifying a content of a stored file (i.e., “**modifying a content of the first file stored on the first client device**”). EX1021, Microsoft Computer Dictionary (5th ed. 2002), 463 (entry for “save” stating “[t]o write data (typically a file) to a storage medium, such as a disk or tape”), 220 (entry for “flush²” stating “[t]o clear a portion of memory. For example, to flush a disk file buffer is to save its contents on disk and then clear the buffer for filling again.”).

158. Sigurdsson also discloses sending the *entire* file once it has been modified. Sigurdsson, ¶ 8 (“Additionally, extracting the content of the first file may include *generating a copy of the first file that retains the first file's original file formatting.*”); *id.* ¶ 28 (“The Client A Content 108, which can include *a copy of the user's file*, is sent automatically to the Server 106.”); *id.* ¶ 54 (“In step 506, the content originating on the client is posted to the server. The content can include *a copy of the document* (or the content of the document) that was saved by the user as

well as metadata that describes information, such as the type of data, its source, when it was saved, and when it was posted to the server.”); *id.* ¶ 65 (“The content can include **a copy of the document** that the user saves as well as metadata that describes the type of data, the client that generated the data, and when the data was last changed.”).

159. Lastly, Sigurdsson discloses that its client devices (and server system) may be “**associated with a user.**” Sigurdsson, ¶ 31 (“The Server 106 can facilitate synchronization of content among the user’s computers.”). Client A is described as, for example, a “user’s home computer” and Client B is described as, for example, a “user’s work computer.” *Id.* ¶¶ 28-29, 55. Figure 4 depicts Client C as a user’s mobile device. *Id.*, Fig. 4, ¶ 27 (“A user may access many computers including, for example, computers at home, computers at work, and mobile computers, such as personal digital assistants (PDAs) and cell phones.”). Clients A, B, and C are associated with the same user. *Id.* ¶ 58 (“For example, a user can store a very large document (e.g., a maintenance manual) on the Client A 102, and at a short time later, the same user empties a digital camera on the Client B 104. The same day, the user may go to work and update a small office memo on the Client C 406.”); *see also id.*, ¶ 2 (“[U]sers want to have access to their digital information whether they are on their home computer, using a laptop at work, or on the road with a wireless personal

digital assistant.”). The Server is also associated with the user because the Server can store and authenticate a user ID for each user. *Id.* ¶ 49 (“The Authentication Manager 334 can provide security features that prevent unauthorized clients from using the Server 106 for storing and requesting content. Authentication can be based on, for example, a system for verifying user IDs and passwords. The Authentication Manager 334 can check the credentials of a client when the client connects to the Server 106. If a client supplies invalid credentials, the Authentication Manager can transmit an error message to the client.”); *id.* ¶ 27 (“The **Server 106** can provide a temporary storage location, or “drop box”, for **a user’s** files, documents, web pages, or the contents of these documents.”).

- e. **[1c]: “receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file”**

160. Claim [1c] is disclosed by and obvious over Sigurdsson in combination with Shappell.

161. For starters, Sigurdsson discloses that its server system “**receive[s], over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file,**” as claimed.

162. As explained for claim [1b], Sigurdsson discloses a collection of client devices connected to a server system over a network. Sigurdsson, ¶¶ 26-27, 69, 112-13. The server system (e.g., Server 106) may receive an updated version of a first file from a client device (e.g., Client A) in response to the user modifying the content of the first file. *Id.*, e.g., ¶¶ 8, 26, 28, 32, 45-47, 54, 63-65, 84-85, 103.

163. Sigurdsson further discloses that the server system may also receive, in addition to the copy of the first file itself, “**first metadata**” associated with that file from the first client device (e.g., Client A) – such as the time that the modified file was saved. Sigurdsson illustrates with an example:

For example, referring to FIG. 1, a user at the Client A 102 may edit and save his resume on his home computer. This action may generate an event that is detected by an event listener. . . . [T]he metadata is a time stamp that specifies when the action is detected. For example, the time stamp can reflect the time that a file was saved ***[M]etadate describing the content, such as the time the resume was saved, can also be transmitted to the Server 106.***

(Sigurdsson, ¶¶ 32-34; *see also id.*, ¶¶ 5 (“The method also includes automatically extracting content from the first file in response to the event using the first client and ***generating metadata to associate with the content[.]***”), 6 (“The method can also include, receiving from a server an indication that the server is configured to transmit the content to a requesting client; and having received the indication and if the peer client is not currently networked to the first client, ***transmitting the content and the***

metadata to the server.”), 33 (“The extracted content can then be used to update the *index that links, or associates, the content with metadata or key words from the content*. In some implementations, the metadata is a time stamp that specifies when the action is detected.”), 35 (“the Server 106 updates database tables with the Client A Content 108 *and any corresponding metadata* describing the Content 108 it received from the Client A 102.”), 54 (“In step 506, the content originating on the client is posted to the server. The content can include a copy of the document (or the content of the document) that was saved by the user as well as *metadata that describes information, such as the type of data, its source, when it was saved, and when it was posted to the server.*”); 65 (“metadata that describes . . . when the data was last changed”);¹⁰ 72-73 (describing “list of the content that was generated” when Client C was offline, such as “metadata describing the file and the time it was accessed may be transmitted to the Server. This metadata may be included in a list tracking content that was generated during the time that the Client C was offline

¹⁰ Although this exemplary metadata in paragraph 65, reflecting when a file was last modified, is disclosed in the context of a “peer-to-peer content flow” (see Sigurdsson, ¶ 65), it would have been obvious to incorporate this type of metadata into embodiments of a client-server content flow, such as the example provided in Sigurdsson paragraphs 72-73 where Client A transfers metadata associated with a large file (e.g., maintenance manual) to the server. Sigurdsson, ¶¶ 65, 72. For example, a POSA would have been motivated to include metadata about when a large maintenance manual (“**first file**”) was last modified so that users know if they have the most up-to-date maintenance instructions.

even though the content is not currently stored on the Server 106.”), 97 (“Additionally, the metadata associated with the content can include expiration dates or other retention rules. This metadata may be transferred along with the content to all of the requesting clients. In this way, the clients that received the content and metadata can delete the content associated with the rules or expiration dates, even if the receiving client does not connect to the network after receiving the content (and therefore never receives the command to delete the content).”).

164. Server 106 **receives over a network, from the first client device** (e.g., Client A), the above-described **first metadata**, along with the updated first file (per claim 1[b]). Sigurdsson, ¶¶ 6 (“***transmitting the content and the metadata to the server.***”), 35 (“Server 106 updates database tables with the Client A Content 108 ***and any corresponding metadata describing the Content 108 it received from the Client A 102.***”); 54 (“In step 506, the content originating on the client is posted to the server. The content can include ***a copy of the document . . .*** that was saved by the user as well as ***metadata that describes information, such as the type of data, its source, when it was saved, and when it was posted to the server***”), 72-73 (list containing generated metadata “***transmitted to the Server***”).

165. Claim [1c] further recites “**the first metadata being assigned a priority that is greater than a second priority assigned to the copy of the first**

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

file,” and claims [1d] and [1i] also recite further “priority”-related limitations. Therefore, for convenience, I will address the interrelated “priority”-related limitations of [1c], [1d], and [1i] together.

166. For context in interpreting these priority-related limitations, the ’823 specification states only: “An embodiment’s directory update system updates at a higher priority than the documents to be synchronized.” ’823, 9:28-29; ’896 provisional, at 17. No affirmative assignment of priority values is described. Nor does the specification explain which component, if any, controls the “higher priority.” Consistent with the non-limiting disclosure in the specification, claim [1c] does **not** recite that “the first metadata being assigned a first priority” must be assigned its priority **prior to** the server system receiving that “first metadata.” On the contrary, claim [1i] expressly provides that that server itself may be the only entity that assigns “the” first priority to the first metadata, which could naturally occur **after** the server receives that metadata. See claim [1i] (“at least one of the server system or the first client device assigns the first priority to the first metadata”). Also, the ’823’s Summary of the Invention (which includes new matter not contained in the purported 2007/2008 priority applications) similarly mentions only the server **transmitting** metadata before transmitting the associated file, with no disclosure of the server **receiving** metadata before receiving the associated file.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

'823, 4:3-18. Nevertheless, to the extent [1c] were narrowly interpreted to require that the “first priority” be assigned to the first metadata **before** the metadata is received by the server system, the prior art discloses and renders that obvious, as discussed below.

167. For the reasons below, under any reasonable interpretation consistent with the specification, Sigurdsson in view of Shappell and Rao disclose and render obvious [1c], [1d], [1h], and [1i], together providing **the server system to automatically transfer the first metadata (received from the first client device) over a network to a second client device prior to transferring the first file, based on a first, greater priority assigned to metadata by a priority assignment configuration on the first client device or server system that assigns a first, greater priority to metadata and a second, lower priority to files.**

168. Under a first interpretation, if these limitations are interpreted to encompass a server system configured to receive metadata from a first device, and then automatically (not user-manually) transfer that metadata to a second client device before transmitting associated files, Sigurdsson alone discloses and renders them obvious (in addition to rendering them obvious with Shappell and Rao, as discussed further below).

169. Sigurdsson discloses and renders obvious that Server **106** receives from

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

the **first client device** a file's generated metadata *before* receiving the file, and then the server automatically transfers that metadata to a **second client device before** transmitting the file. Sigurdsson acknowledges the concern of limited bandwidth available for data transmissions between clients and the server. Sigurdsson, ¶ 58 ("Since bandwidth on the Server 106 and on the Clients 102 and 104 may be limited, and user actions can transmit or request large amounts of data at one time, content sharing can fall behind demand."). To address this concern, Sigurdsson discloses the concept of assigning varying priorities to different types of files and transmitting the higher-priority files first. *Id.* ("Although the Server 106 can receive content for all three data items in the order that the user accessed them, ***the Server 106 can also deal with the data items by assigning it priorities. The Server 106 can handle the higher priority data items first, and save the lower priority data items for later.*** In one implementation, the priorities are based on attributes such as their size, type or age, which are each associated with a score. A lower combined score can be assigned a higher priority. ***For example, a small document (e.g., an office memo) would receive lower score based on size, thus a higher priority. Similarly, a large document (e.g., a maintenance manual) and digital photos would receive a higher score (and a lower priority).*** . . . A data item's type [e.g., metadata or file data] can also be used to calculate its priority, since, for example, an office environment can

place a higher priority on work-related data items, such as documents generated by Microsoft Word™. In some implementations, the priority algorithm is used by each of the clients to determine when to transmit content from that client to the server or another client.”).

170. To implement such file transmission prioritization, Sigurdsson discloses a “priority algorithm” (discussed below), optionally implemented in the system on the clients and/or Server 106, which prioritizes the order in which files are transferred. Sigurdsson, ¶¶ 58-60, 72. Higher-priority (e.g., smaller and/or newer) files may be transferred before lower-priority (e.g., larger and/or older) files. *See id.* As a result of the priority algorithm, Server 106 can first receive from the first client device the **metadata** associated with a low-priority file that Server 106 has not received. Sigurdsson describes: “The content of this file may not be transmitted to the Server 106 ***because of the execution of a priority algorithm*** (as discussed earlier). ***Instead, metadata describing the file and the time it was accessed may be transmitted to the Server 106. . . .*** even though the content is ***not currently*** stored on the Server 106.” *Id.*, ¶ 72; *see also id.* ¶¶ 58 (“The Server 106 can handle the higher priority data items first, and ***save the lower priority data items for later.***”), 96 (“If the content is not within the Storage 722, the Server 106 may request it from the client that originated the content. After the server receives the

requested content, it may transmit it to the Client A 102.”). And although it does not appear to be a requirement of the claim, this “metadata describing the file and the time it was accessed” obviously includes metadata generated from the user modifying the file content (which would change the file’s description and last-accessed time), matching the “metadata describing the content, such as the time the [file] was saved” generated when a user modifies a file. *Id.*, ¶¶ 34, 72.

171. Sigurdsson further describes its “priority formula” for assigning priorities to files as follows: “one implementation can be stated as $1000 * \log N(s) \sqrt{(t/K)/B_{\text{type}}}$, which is explained as follows: a data item would be assigned a base priority of 1000. The base priority is multiplied by the square root of the scaled time at which it was generated. This serves to significantly raise the priority of a very new data item. The priority is multiplied by $\log N(s)$ (the logarithm with the base N of s), where N is a constant and s is the size of the file in kilobytes. That way, small files would get priority over large files, yet large files do not get pushed too far to the back of the priority list since the factor is not linear. Finally, a constant ‘boost factor’ B can be applied based on the type of the data item. The boost factor is 1 by default, but can be different for different types of data items (e.g., $B_{\text{ms-office}}=2$), where ms-office signifies a Microsoft Office™ document.” Sigurdsson, ¶ 59.

172. Metadata received by Server 106 from a **first client device** may be transmitted by Server 106 to a **second client device** before Server 106 transfers the associated file. Specifically, the received “**metadata** describing the file and the time it was accessed . . . **may be included in a list tracking content** that was generated” during a period of time when another client (e.g., Client C) was offline “even though the content is not currently stored on the Server 106.” Sigurdsson, ¶ 72. Once that client (e.g., Client C) comes back online, it requests and receives from Server 106 that “list of missing content” including the metadata. Sigurdsson, Fig. 6 (steps 612, 614, 616), ¶¶ 69-73. That client can then use the list to obtain the missing files, which may eventually be transferred by Server 106, such as if no other sources are available or Server 106 is a better source for obtaining the file. Sigurdsson, ¶¶ 79-80.

173. Sigurdsson also provides a second, complementary teaching—setting aside Sigurdsson’s optional “priority algorithm”—of how Server 106 **prioritizes transmitting metadata to the second client prior to transmitting the associated file**. As noted, Sigurdsson explains that when a client device that was offline comes back online, it can request a list of content from Server 106 that it is missing due to having been offline. Sigurdsson, Fig. 6 (e.g., steps 612, 614, 616), ¶¶ 69-73. The list of missing content includes metadata about the content, including the types of

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

metadata discussed above generated by the user modifying a file at the first client device (and received from that device), such as time ranges associated with “accessing” the content. Sigurdsson, ¶ 73. After the now-online client receives that list, it can use it to download the associated modified file(s) from Server 106. Sigurdsson, Fig. 6 (e.g., steps 620, 626, 628), ¶¶ 74, 79-81. Accordingly, the client device’s request for a list of missing content **configures Server 106 to prioritize transmitting the first metadata received from the first client device to the second client device before transmitting the file.**

174. Under a second interpretation, if the priority-related limitations of [1c], [1d], and [1i] are interpreted to require affirmative, categorical assignment of greater/lesser priorities to metadata and files (which the ’823 specification does not describe), they are additionally rendered obvious by Sigurdsson with Shappell and Rao.

175. As noted, Sigurdsson teaches that the overall “system 400” (including Server 106 and/or client devices) optionally assigns “priority” to items using a “priority algorithm,” to address the concern that limited available network bandwidth may cause “content sharing [to] fall behind demand.” Sigurdsson, ¶ 58. For example, “[a]lthough the Server 106 can receive content for all three data items in the order that the user accessed them, **the Server 106 can also deal with the data**

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

items by assigning it priorities. The Server 106 can handle the higher priority data items first, and save the lower priority data items for later.” *Id.*, ¶¶ 58-59 (describing priority scores assigned to documents/files based on attributes such as “size, type or age”). Sigurdsson also states: “In some implementations, the priority algorithm is used by *each of the clients* to determine when to transmit content from that client to the server or another client.” *Id.*, ¶ 58. Therefore, it would have been obvious to use Sigurdsson’s priority algorithm to assign transmission priorities at a client to prioritize transmission of metadata prior to file data before the client sends the metadata to the server. *See Id.*, ¶ 72 (disclosing that Client A may prioritize sending a large file’s **metadata** to Server 106 above sending the large file itself).

176. While Sigurdsson does not expressly describe assigning priority values to metadata, Sigurdsson in view of Shappell and Rao renders obvious a predictable variation where the priority algorithm (**priority assignment configuration**) in the system, configured on the first client and/or Server 106, categorically assigns greater priority to metadata and lower priority to files, thereby causing metadata to transfer prior to the transfer of the associated files based on the greater priority, to advantageously inform clients about updated files.

177. Sigurdsson itself motivates this modification. The purpose of Sigurdsson’s priority algorithm is to prioritize “data items” based on, for example,

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

“size, type or age,” in view of potential limited bandwidth, as “content sharing can fall behind demand.” Sigurdsson, ¶ 58. Sigurdsson teaches assigning a greater priority to ***smaller*** and ***newer*** data items—precisely the attributes a POSA would recognize as applicable to ***metadata*** (typically containing much less data than the file itself) for an ***updated*** file—as well as the ***type*** of “data item,” which obviously would differentiate metadata from file contents. *Id.*, ¶¶ 58-59. A POSA would also appreciate the natural harmony between the priority algorithm and Server 106 prioritizing the “list of missing content” (containing metadata) to inform clients about new content at Server 106, as well as Sigurdsson’s teaching that “Server 106 can ***handle the higher priority data items first, and save the lower priority data items for later.***” *Id.*, ¶ 58. From these teachings, a POSA would have found it obvious to simply configure the priority algorithm to assign greater priority to metadata, and lower priority to the associated files, advantageously and predictably causing transfer of metadata before the associated files, accounting for Sigurdsson’s “bandwidth” considerations in transferring files. *Id.*, ¶¶ 60, 79.

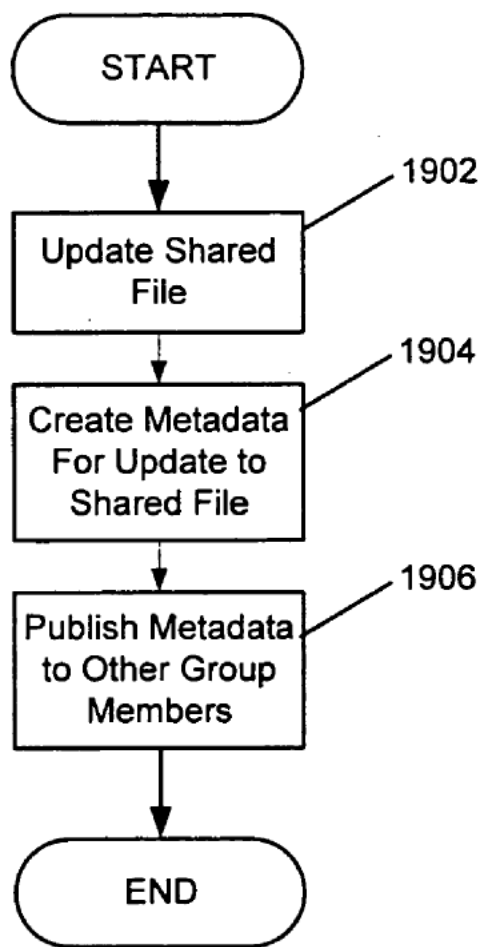
178. **Shappell** and **Rao** both further disclose and motivate this obvious modification to Sigurdsson.

Like Sigurdsson, **Shappell** teaches synchronizing files across multiple computers using metadata. Shappell, ¶ 9. Shappell discloses “shared spaces” in

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

which files may be shared among group members' computers. *E.g., id.; id.*, ¶ 47.

Shappell teaches that “[w]hen a **modification of a file**” occurs, “**metadata is transmitted to all members**” of the group. *Id.*, ¶¶ 81, 76, Fig. 19 (steps 1902 (“Update Shared File”), 1904 (“Create Metadata For Update to Shared File”), 1906 (“Publish Metadata to Other Group Members”).



(Shappell, Fig. 19.)

Shappell teaches that dissemination of metadata to all group computers serves

an important function: “[t]o **inform a user** of a group shared space file that is not present locally” by displaying an indicator at each computer “**to denote a ‘missing’ file that is available on another system.**” Shappell, ¶¶ 79, 76 (the “metadata are sufficient to render an icon of the shared file and may include date, time, thumb-nail information, name size, and optionally the source of the information”). In one illustration, the indicator is an icon that is “ghosted or semi-transparent.” Shappell, ¶¶ 79, 81.

If a version of the file already exists locally, the receipt of metadata causes the computer to either “remove the local copy of the out-of-date file,” which “will change the visual representation of the file to a ‘missing’ file,” or alternatively obtain “the updated version of the file” if “automatic replication” is enabled. Shappell, ¶ 81, Fig. 20 (steps 2002, 2004, 2010, 2006); *id.*, ¶ 62 (describing file replication alternatives), ¶¶ 77-78 (describing types of metadata).

179. Thus, Shappell teaches that a file’s metadata—such as file’s “time”—is transferred first (with a higher priority) from an originating computer to all connected group computers **before** transferring a copy of the file itself, to beneficially facilitate user-interface display of the file’s availability.

180. **Rao** provides naturally complementary teachings to categorically prioritize synchronizing metadata before files between server and client, to

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

advantageously enable functions at the client. Like Sigurdsson, Rao describes file synchronization between a client and a server over a network. Rao, ¶¶ 26, 37. Rao discloses synchronizing files from a server to a client in two phases: first the metadata, and then the “raw data” of the files themselves. Rao, ¶ 63 (“In the ***first phase, metadata is synchronized between a server and client*** while the raw data remains on the server system. The client system, when connected or disconnected, may use and manipulate the metadata for any purpose whatsoever, including selecting one or more raw data files that are to be available on the client system. ***The second phase of synchronization is to synchronize the selected raw data to the client system.***”). Much like Shappell, Rao teaches that prioritizing metadata enables functions using the metadata beneficial to “the user of the client” receiving the metadata, such as to “display” information about files. Rao, ¶¶ 63 (“the client system . . . may use and manipulate the metadata for any purpose whatsoever”), 43 (“Because the metadata relating to file 140 is contained in file backed item 132, certain functions and operations may be available to the user of the client system 104,” such as “to display . . . audio tracks . . . and perform other operations using only the metadata”).

Motivation to Combine Sigurdsson with Shappell and Rao

181. It would have been obvious to combine Sigurdsson with the metadata-

priority teachings of Shappell and Rao, predictably resulting in Sigurdsson's synchronization system where a priority algorithm at the first client and at Server 106 assigns a first, greater priority to metadata and lesser priority to files, resulting in the metadata transferring first (based on the first priority) before transferring associated files.

182. Sigurdsson, Shappell, and Rao are analogous references in the same field as the '823, directed to synchronizing files and associated metadata. All three references are directed to techniques for effectively synchronizing modified versions of files between devices over a network. And all three references are reasonably pertinent to the problems faced by the inventor of the challenged patent for improving upon existing file synchronization systems.

183. The combination would have straightforwardly combined prior art elements according to known techniques to yield a predictable improvement to Sigurdsson's system, improving the Sigurdsson system in the same way that the relevant features enhance Shappell's and Rao's disclosed systems, by automatically transmitting metadata as a priority based on file changes to timely notify users that an updated version of a file is available (Shappell) and assigning higher priority to that metadata over a lower priority assigned to files (Rao).

184. A POSA also would have been motivated to consider both Shappell and

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Rao given their originations with leading file-management vendor Microsoft, as Sigurdsson likewise contemplates standard “home” and “work” computers with Microsoft software. Sigurdsson, ¶¶ 27, 42, 45, 58, 60.

185. Furthermore, a POSA would have been motivated to consider additional ways to address Sigurdsson’s stated bandwidth concerns. Sigurdsson, ¶¶ 51, 58 (with limited bandwidth, “content sharing can fall behind demand”), 60. A POSA would have recognized that Sigurdsson’s system gives rise to a potential gap in synchronicity between versions of a synchronizing “very large document” over a slow network connection, without notifying the user. Sigurdsson, ¶ 58. Indeed, as Rao cautions, “synchronizing” a very large set of files “may take many minutes or even several hours, depending on the throughput of the communication path . . . among other things.” Rao, ¶ 48. As an illustration, the user of a synchronizing client device (e.g., work computer) in Sigurdsson may work on modifying the large file. When the user subsequently accesses another synchronizing client device (for example, home computer) to continue work on that file, the user would benefit from being able to quickly and visually determine whether or not the version of the file on the home computer is current, before opening the file to work on it. In view of this potential deficiency in Sigurdsson, a POSA would have been motivated to consider Shappell and Rao for a solution that

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

immediately transmits metadata generated by modifying the file to the user's other client devices (including through Server 106, using its priority algorithm, to account for other client devices that may be currently offline), advantageously leveraging the metadata for "any purpose" such as to "***inform a user***" that an updated file is available. Shappell, ¶ 79 (motivating use of metadata "[t]o ***inform a user*** of a group shared space file that is not present locally" by displaying an indicator "***to denote a 'missing' file that is available on another system***"); Rao, ¶¶ 63 (motivating that "***the client system . . . may use and manipulate the metadata for any purpose whatsoever***"), 43 (motivating that the metadata enables that "***certain functions and operations may be available to the user***" of the receiving client, including to "display" information regarding files).

186. More generally, the practice of prioritizing the transfer of metadata before its associated content, and the benefits of doing so, would have formed part of the background knowledge of a POSA. By the early 2000s, well-known commercially available software prioritized metadata transfer for similar reasons. For example, as explained in Boyce, Microsoft Outlook downloaded headers for email messages—which comprise metadata such as a message's subject, date, sender and recipients—before downloading the messages themselves. Boyce, 366-370. Boyce echoes the user interface benefits noted above, explaining that the

“advantage” of this functionality “is the ability to work with the messages headers of waiting messages without having to download the messages themselves. You can simply connect to the e-mail server, download the headers for new messages, and disconnect. You can then take your time reviewing the message headers to decide which messages to download, which ones to delete without reading, and which ones to leave on the server for later.” Boyce, 368; *see also id.* (“When you want to process messages selectively”). Boyce provides a relevant example of a large message included among other messages on the email server, explaining that the functionality to prioritize obtaining email headers “is extremely useful when you’re pressed for time but have a message with a large attachment waiting on the server. You might want to retrieve only your most critical messages without spending the time or connect charges to download that message and its attachment.” Boyce, 366. “To accomplish this, you can connect with remote mail and select the messages you want to download, leaving the one with the large attachment on the server for a less busy time when you can download it across the network or through a broadband Internet connection.” Boyce, 366; *see also id.*, 196 (describing functionality to configure system to only download message headers for messages larger than a specified size).

187. As another example of commercially available software, Boyce explains that Microsoft Outlook Express included functionality to send and receive

newsgroup messages. Boyce, 307-308. Boyce describes how Outlook Express would first download the headers—information such as message subject, size and date—for newsgroup messages before downloading the messages themselves. Boyce, 307-308 (“After you select a newsgroup, Outlook Express downloads a certain number of messages headers from the newsgroup by default. . . . [Reading a newsgroup message] can be as simple as opening the newsgroup and clicking the message header to download the message and view it in the preview pane.”).

188. A POSA would have readily appreciated that the advantages of prioritizing metadata transfer that Boyce describes in the context of email and newsgroup messages—as well-known ways of sharing files and other content among users and computer systems—would have similarly applied to the file sharing and synchronization systems described by Sigurdsson, Shappell, and Rao.

189. A POSA would have reasonably expected success in implementing this modified Sigurdsson system, as Sigurdsson already discloses metadata and file transfer using standard, familiar computer, OS, and network components. A POSA would not have been dissuaded by Shappell’s primary embodiment being peer-to-peer rather than client-server. A POSA would have understood that the relevant features of Shappell relating to transfer of metadata and graphical indication of file availability are independent of the type of network architecture as between peer-to-

peer or client-server. While Shappell's exemplary embodiment is "server-less" (e.g., Shappell, Abstract, ¶ 7), Shappell does not specifically disparage a client/server implementation, and proclaims that the "type of network . . . is not critical" and is only "preferably" peer-to-peer, and suggests also transferring "indirectly" between user machines such as would be provided by a client/server architecture (Shappell, ¶ 40). Similarly, a POSA would not have been dissuaded by Shappell's statement that, in its exemplary Figure 2 embodiment, "[e]ach user device will **typically** be used by or associated with a different user" (*Id.*, ¶ 40), given the compelling applicability of Shappell's relevant teaching to improve Sigurdsson's system as explained above.

190. A POSA would have been further motivated to look to references like Shappell, given that it was assigned to Microsoft Corporation, in view of the popularity of Microsoft Windows at the time and a POSA's motivation to study the various file sharing and synchronization solutions invented by Microsoft that are disclosed in, e.g., Shappell (which indicates on its cover page that it was assigned to Microsoft). As noted above, a POSA would have known that Microsoft was a significant vendor at the time in file management solutions, including file sharing solutions such as those described in Shappell.

191. A POSA reviewing Sigurdsson would have been motivated by

Sigurdsson's disclosure related to prioritizing different kinds of files and different sizes of files to consider other solutions aimed at addressing the bandwidth concerns addressed in Sigurdsson. Sigurdsson, ¶ 58. One of these solutions would have been Shappell's solution of sending metadata associated with files first (before sending the actual file data). A POSA would have known that metadata of a file is often smaller in size than the file itself—particularly when the file is large—and could therefore be transmitted more quickly and with lower bandwidth consumption than the file itself. A POSA would have understood the benefit of looking for solutions like Shappell's to allow users to see which files, and particularly which modified updated versions of files, are available via metadata before actually downloading the files themselves in their entirety.

192. Furthermore, Sigurdsson discloses a very similar embodiment to Shappell's related to prioritization of sending metadata before file data. Sigurdsson, ¶ 72 ("The content of this file may not be transmitted to the Server 106 ***because of the execution of a priority algorithm*** (as discussed earlier). ***Instead, metadata describing the file and the time it was accessed may be transmitted to the Server.*** This metadata may be included in a list tracking content that was generated during the time that the Client C was offline even though the content is not currently stored on the Server 106.""). A POSA would have recognized in view of Shappell, however,

that Sigurdsson does not expressly address the issue of notifying users that an updated version of a file is available through the network, and that Shappell teaches and motivates a solution to that issue.

193. A POSA would have been further motivated to look to similar solutions, like that disclosed in Shappell, in view of Sigurdsson's teachings related to a customizable "priority formula" for determining when to send different kinds and sizes of files. Sigurdsson, ¶ 59. Sigurdsson teaches transmitting different types and sizes of data content (files) with different priorities, and a POSA would naturally find the same considerations to apply to Shappell's teachings of transmitting metadata (of small size) for notification purposes prior to transmitting copies of files. As explained above, separately transmitting metadata to a receiving client device would enable the receiving client device to visually indicate the status of the updated file – for example, to indicate that an updated version was available but not yet downloaded as a result of the priority formula. The user could then opt to download the file as needed, as Shappell describes, or have the client device eventually automatically download the file in accordance with the priority formula.

194. A POSA would have had at least a reasonable expectation of success in implementing the modified system of Sigurdsson in this manner. Sigurdsson already discloses the presence of metadata and the use of standard personal computers and

operating systems that were very familiar to a POSA. Transmitting metadata and providing a graphical availability indication as taught by Shappell would have been well within the grasp of a POSA, as well as implementing the modified system of Sigurdsson accordingly, using well-known computer software and networking knowledge and design principles. The references provide ample detail such that a POSA would have been able to implement the modification without undue experimentation.

- f. **[1d]: “automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device”**

195. Sigurdsson in combination with Shappell and Rao render obvious element [1d] as explained for claim [1c] and further below. As explained for [1c], the references render obvious Sigurdsson’s system **automatically transferring the first metadata over a network to a second client device before transferring the first file to that device.**

196. As explained above for claim [1b], Sigurdsson teaches multiple clients (e.g., Clients A-C) **associated with the same user** and communicating with Server 106 **over a network.** (Sigurdsson, ¶¶ 27-29, 55, 58, Fig. 4.)

- g. **[1e]-[1f]: “wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and (ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device”**

197. Sigurdsson in view of Shappell discloses and renders obvious [1e]-[1f]. To start, as explained for [1c] and [1d], Sigurdsson in view of Shappell and Rao disclose and renders obvious that Server 106 **transfers the first metadata to the second client device before the copy of the first file.**

198. Sigurdsson discloses that its server and clients may include graphical user interfaces: “The Input/Output Devices 940 provides input/output operations for the System 900. In one implementation, the Input/Output Devices 940 includes a keyboard and/or pointing device. In another implementation, the Input/Output Devices 940 include a display unit for displaying graphical user interfaces.” Sigurdsson, ¶ 108; *see also id.* ¶ 111 (“To provide for interaction with a user, the features can be implemented on a computer having a display device such as a CRT (cathode ray tube) or LCD (liquid crystal display) monitor for displaying information to the user and a keyboard and a pointing device such as a mouse or a trackball by which the user can provide input to the computer.”). A POSA would

have found it obvious to consider how to indicate to users the current state of files being synchronized using Sigurdsson systems and methods and would have been familiar with Microsoft Windows as a well-known operating system including graphical user interface components for indicating the state of file transfers and sharing, as disclosed in Shappell, for example. Thus, a POSA would have looked to references like Shappell to pick up where Sigurdsson leaves off and consider Shappell's more detailed disclosures about obvious options for user interfaces, including how to indicate to users the state of files before, during, and after synchronization via graphical user interfaces.

199. Shappell discloses a **“graphical availability indication”** that, **“before the copy of the first file is transferred to the second client device,”** is **“presented at the second client device based on the first metadata.”** As discussed for [1c], Shappell teaches that a computer's receipt of metadata caused by a file modification causes the display of a “ghosted or semi-transparent” icon. Shappell, ¶¶ 79, 81, Figs. 19-20. The ghosted icon discloses one example of the claimed **“graphical availability indication,”** as it graphically indicates that an “updated version of the file” is available. Shappell, ¶¶ 79, 81.

200. As explained for [1c], Shappell renders obvious using the **“first metadata”** described in the foregoing limitations as originating from Client A in

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Sigurdsson to cause the presentation of Shappell's exemplary **"graphical availability indication"** at Client C (**"second client device"**) based on the **"transfer of the first metadata to the second client device,"** predictably improving Sigurdsson's system with a convenient and user-friendly notification displayed at Client C to inform the user of Client C that an updated file is available from the server. Shappell, ¶¶ 76, 79, 81.

201. Shappell further renders obvious that the **"graphical availability indication"** **"is presented proximate a file icon representing the first file on a user interface of the second client device"** (claim [1f]). Beyond the exemplary **"graphical availability indication"** (the "ghosted" semi-transparent icon) discussed above for [1e], Shappell renders obvious alternative availability indications that obviously would be presented **proximate a file icon representing the first file.**

202. For context, Shappell teaches the use of the well-known Microsoft Windows OS and motivates that its shared space **"is presented through a graphical user interface with the look and feel of existing file system features,"** such as **"Windows Explorer-style dialog boxes,"** with ***"graphical user interface presentations and controls that are included as part of an existing file system, or at least an intuitive extension of existing file system operations."*** Shappell, ¶¶ 42, 45, 47, Fig. 3 (**"Windows Core OS" layer 302**).

203. Windows provided a plethora of user-interface features, including numerous graphical displays of information about a file presented **proximate a file icon**. Shappell itself teaches several examples, and a POSA would have understood in view of Shappell’s stated GUI goals that those are merely examples among many GUI design-choice options, both available in Windows and “intuitive” extensions.

204. For example, Shappell describes that to “enable the user to discern what content is new or has been recently altered, *a ‘new’ glyph may be presented on folders and files that are new or have been recently changed.*” Shappell, ¶ 80 (“This glyph provides a visual indicator of new content.”). A POSA would have understood this “new” glyph presented “on” “**recently** changed” files is and obviously would be presented **proximate the file icon**, as an additional overlay on the icon that may optionally extend beyond the underlying icon’s visible graphic, which may depend on the size/space of the underlying icon.¹¹ For example, Barber (EX1024, U.S. Pat. No. 5,923,325) discloses a graphical availability indicator (boxed in red) associated with and proximate to a file icon (boxed in green) that does not overlap with the file icon:

¹¹ A POSA would have understood “proximate” an icon to encompass overlays of distinct graphics, including those that may overlap at least in part with the underlying icon. *E.g.*, EX1023, European Patent Application Publication No. EP 1,621,990 A2 to Kitamaru, et al. (describing that “the window 5113 is displayed **near** the icon 5105” where window 5113 partly overlaps with icon 5105).

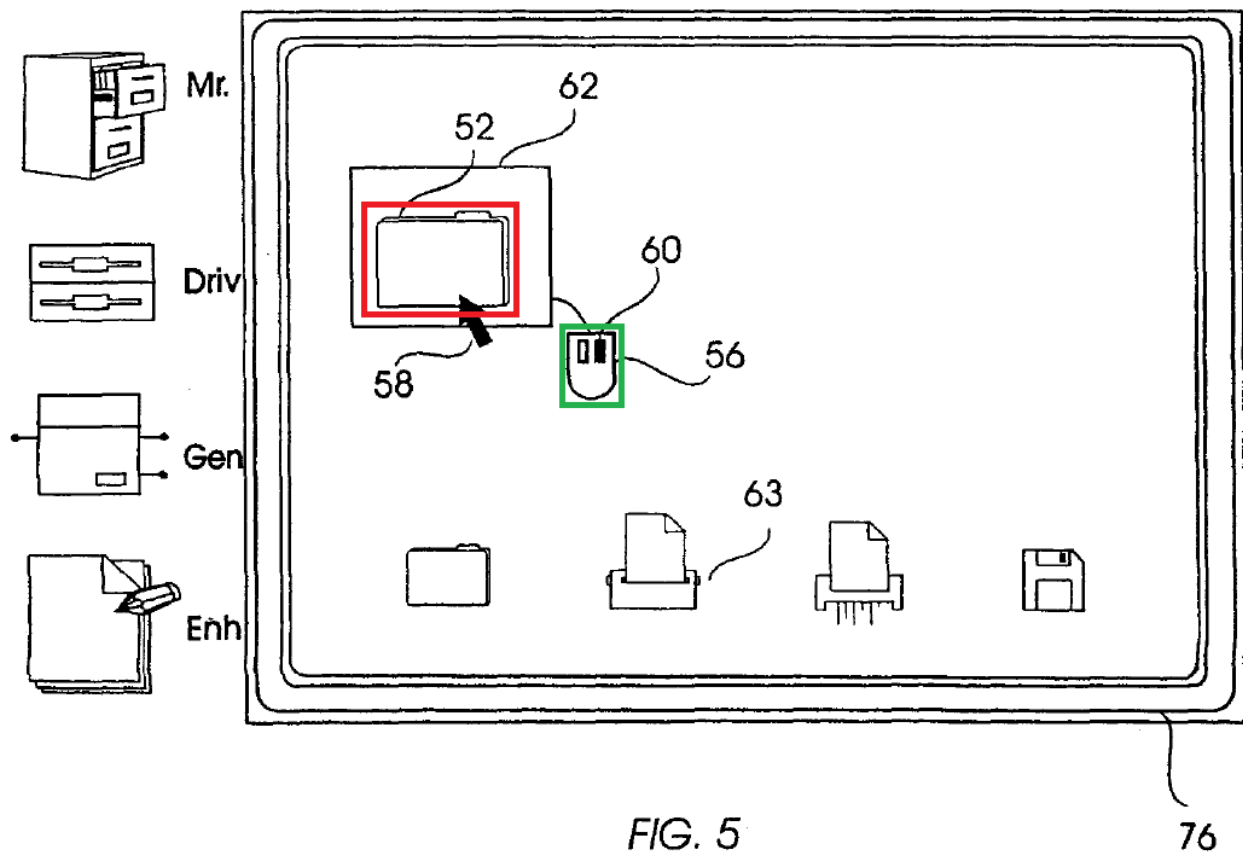


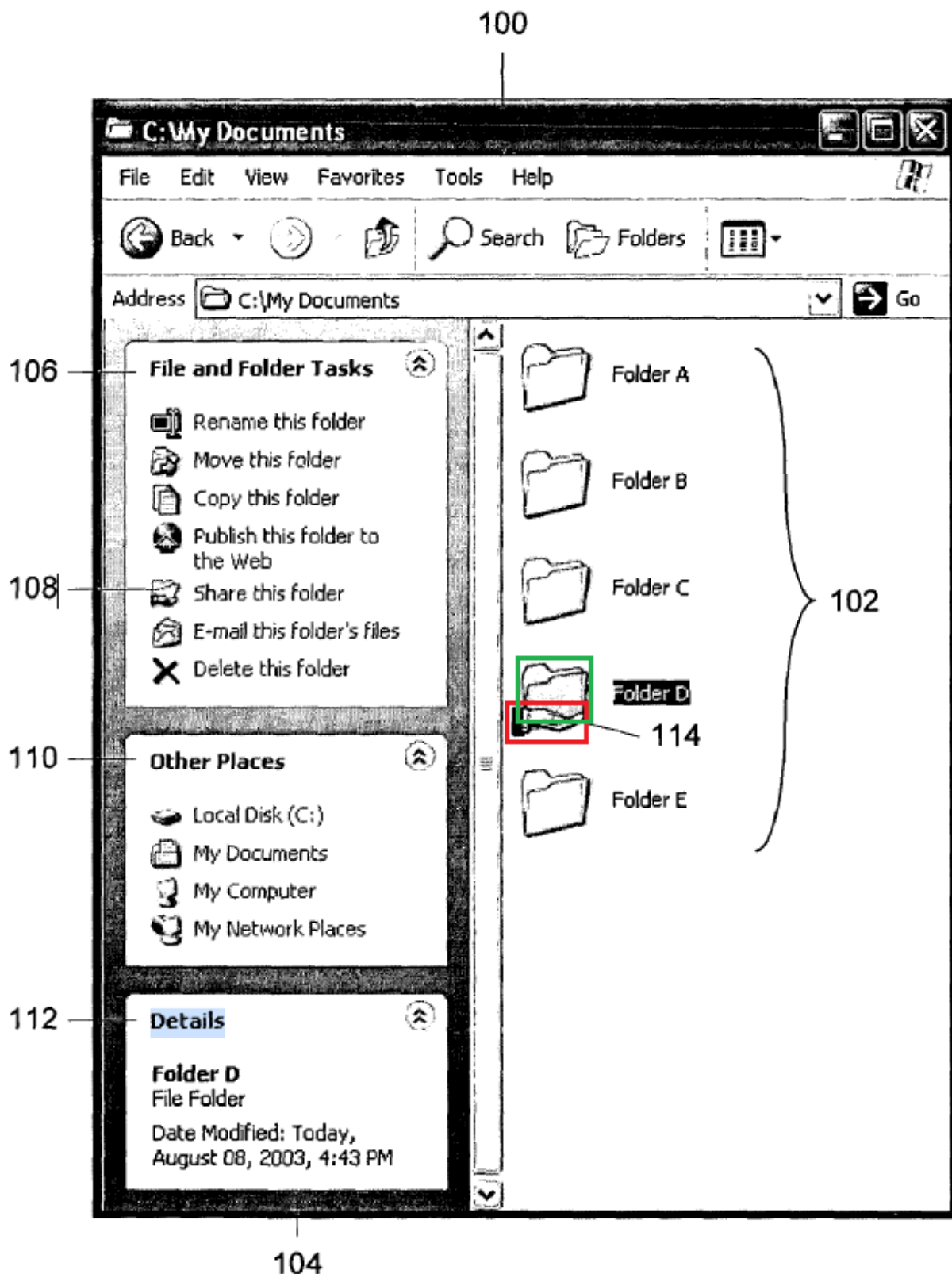
FIG. 5

76

Barber, Fig. 5 (annotated); *see also id.*, 6:6-9 (“In FIG. 5, the object 52 is selected for printing. The first help image illustrates that a mouse 56 is manipulated so as to move a cursor, represented by an arrow 58, onto the object 52.”). This is one obvious variant of a graphical availability indication that is proximate (i.e., near) a file icon but not overlapping the file icon. Another obvious variant would be to overlay a graphical availability indicator such that it slightly overlaps with an underlying file icon that is proximate to the graphical availability indicator, as shown in the following example from Figure 1 of Moromisato, showing a hand overlay graphical availability indicator (boxed in red) overlapping an associated file icon (boxed in

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

green):



Moromisato, Fig 1 (annotated); *see also id.*, 3:15-17 (“When a selected folder is designated as shared, a ‘hand’ overlay is applied to the icon that represents the folder as shown at 114 in FIG. 1.”).

205. As another illustration, where a “missing” file is currently unavailable (for example, its hosting system is offline), “a display icon 1506 is presented as a ghosted icon that includes a small image of a red ‘X’ **or other suitable visual indicia** that the file is unavailable to the user.” Shappell, ¶ 79. The red “X” is presented proximate the file icon. Considered as a whole, Shappell’s teachings of the “new” glyph indicating files that “have been recently changed,” red “X” indicating file **unavailability**, and suggestions to consider “other suitable visual indicia” and “intuitive” GUI extensions, would have prompted a POSA to consider analogous icon-proximate indications for the **availability** of a recently modified “missing” file. The “ghosted” icon **1504** is semi-transparent to indicate a “missing” file that has been modified on another computer, but that “ghosted” icon itself would not inform the user whether the “missing” file was “**recently** changed” or not. Shappell, ¶¶ 79, 80, 81. A POSA would have considered the “new” glyph a trivially obvious design-choice to present proximate the “ghosted” icon for “**recently** changed” “missing” files (for example, using a time threshold to limit the duration of display of such a graphic based on recency), leveraging the “modification date” in the metadata.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Shappell, ¶¶ 78, 81. The glyph would intuitively inform the user, using a GUI feature already present in Shappell’s system: a “new” glyph on the icon of “recently changed” downloaded files and a “new” glyph on the “ghosted” icon for recently modified “missing” files available for download, disclosing **a graphical availability indication proximate a file icon**. Shappell, ¶¶ 78, 81.

206. Shappell’s exhortations to exploit and intuitively extend Windows GUI features would have motivated numerous other **icon-proximate indicators of file availability** obviously supplementing or replacing Shappell’s “ghosted” icon. Numerous icon-proximate indicators were present in Windows for consideration. For example, Shappell describes that “the user may right-click an icon” to “present a graphical interface such as Context Menu 1700” shown in Figure 17. Shappell, ¶ 83, Fig. 17. In this familiar Windows feature, a user right-clicks a file icon, prompting display proximate the icon of a menu of options regarding the file. Shappell does not describe what would happen if a user right-clicked a “ghosted” icon indicating file availability, leaving it to the designer to populate the pop-up “Context Menu” menu options. A POSA would have found it obvious that the pop-up menu would **indicate file availability proximate the icon**, such as displaying a “Download” menu option caused by the receipt of the metadata. In fact, another prior art Microsoft file-synchronization patent (Moromisato) describes precisely

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

such a pop-up “Download” menu option, proximately indicating availability to
“Download” a file. Moromisato, 15:36-43 (a “menu with appropriate actions would
then appear,” including “Download”).

207. A POSA similarly would have been motivated to exploit “[t]he
‘Details’ view in Windows Explorer [that] displays columns of information about
each file in the file system” (Moromisato, 17:14-21), as another obvious preexisting,
convenient, and user-friendly Windows feature to graphically indicate, **proximate a
representation of the “missing” file**, the significant information of which files are
available for download—again, illustrated by Moromisato (*see id.*), complementing
Shappell’s own suggestion to use “Windows Explorer-style” displays (Shappell,
¶ 42).

208. A POSA also would have considered obvious, as described in
Moromisato, that the icon “that represents that file is ***provided with an icon overlay***”
that “can be similar to ***the ‘hand’ overlay 114 (FIG. 1) displayed by Windows XP
Pro*** file system when a folder is shared over the network.” Moromisato, 17:3-14,
15:14-22 (where synchronized “***file downloading is deferred to a later time***, that file
is represented in the OS file system as a ‘stub’ file that is very similar to a Windows
shortcut,” with “the same icon as the target file, but it ***may have an overlay of some
sort***”), Fig. 1. This familiar Windows “hand” overlay for a network-shared resource

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

discloses another obvious **graphical availability indication proximate a file icon** to be presented proximate a ghosted (or alternatively non-ghosted) icon for a “missing” file that is available for download. The classic Windows “shortcut” icon overlay (or a similar adjacent indicator) also would have been another trivially obvious choice to indicate, **proximate the file’s icon**, that the file is not present locally but is available for download. Moromisato, 15:14-22 (where synchronized “**file downloading is deferred to a later time**, that file is represented in the OS file system as a ‘stub’ file that is **very similar to a Windows shortcut**,” with “the same icon as the target file, but it **may have an overlay of some sort**”).

209. Yet another obviously well-suited, familiar Windows feature to supplement or replace Shappell’s “ghosted” icon was “Infotips”—a pop-up graphic “to display extended information” **proximate a file icon** when mousing over it, again naturally applying Shappell’s motivation to use existing Windows GUI features to provide information about remotely-located files. Moromisato, 17:22-29 (illustrating “infotips” in Windows for displaying information about “synchronized files”).

210. Another obvious “intuitive” extension, inspired by the **negative-connotation** red “X” overlay for **unavailability** of a “missing” file, would have been a **positive-connotation** overlay proximate the “ghosted” icon for **availability** for

download, such as a green checkmark (✓).

211. The motivation to combine Sigurdsson and Shappell above in [1c] applies with full force here. Sigurdsson contemplates standard “home” and “work” computers with Microsoft software but does not expressly describe GUI features. Sigurdsson, ¶¶ 27, 42, 45, 58, 60. Shappell picks up where Sigurdsson leaves off, motivating a POSA to consider Windows and Windows-extending GUI features to improve upon Sigurdsson with Shappell’s teachings to graphically indicate to Sigurdsson system users the existence and status of modified files, with full expectation of predictable success.

- h. [1g]: “wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device”

212. Sigurdsson in combination with Shappell discloses and renders obvious claim [1g]. As discussed above for claims [1e] and [1f], Shappell’s disclosed and obviously motivated indicator for an available “missing” file “**indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded.**” Shappell’s indicator of a “missing” modified file (including as obviously implemented proximate a file icon, as per claim [1f]) represents that the file is “**available**” to be downloaded *from another*

system to the computer displaying the indicator. Shappell, ¶¶ 79, 81, 62. In Sigurdsson’s client-server system, that “other system” is Server 106. Sigurdsson, ¶¶ 73-74, 79-80 (explaining that Client C can determine that the missing content is available from the Server and download it from the Server). The same motivations to combine Sigurdsson with Shappell apply as discussed above.

213. This could occur, for example, when Client A completes a full synchronization iteration by first transmitting its metadata to the server and then transmitting an updated file to the server before Client C (which may have been offline) begins a synchronization iteration and subsequently downloads the metadata describing the file that was updated at Client A before downloading the actual file itself. Sigurdsson expressly contemplates this scenario in paragraphs 73, 74, and 79. As discussed in paragraph 73, Client C may be offline and metadata may be transferred from Client A to the server before a copy of the file associated with the metadata, pursuant to the priority algorithm. In paragraph 79, Client C may request a copy of the file from Server 106. A POSA would have understood from paragraph 79 that a copy of the file **must have been** transmitted by Client A to the server subsequent to the transfer of the metadata. As discussed in paragraph 74, Client C may request the metadata and receive it as part of the list of missing content provided by the server to Client C when it comes back online. An obvious variant of this

execution flow would have been for the actual file to have been uploaded to the server sometime before Client C requests and receives the metadata.

214. For example, Client C could be offline for an extended period of time, during which the transmission priority algorithm executed on Client A could work its way through until it sends a copy of the actual file to the server for subsequent transmission to Client C when it comes back online. Sigurdsson, ¶ 83 (“The Server 106 serves as a temporary storage location that can facilitate synchronization between the user’s clients.”). Thus, when Client C finally comes back online, it executes the functionality described in paragraph 74 to obtain the list of missing content from the server, including the metadata, **after** the file associated with such metadata has already been transferred from Client A to the server while Client C was offline.

215. Moreover, even if the file is not immediately stored on the server, Sigurdsson explains that it can readily be obtained by the server on-demand when requested. Sigurdsson, ¶ 96 (“If the content is not within Storage 722, the Server 106 may request it from the client that originated the content.”). Thus, even a file that is not yet on the server is available for download from the server, given that the server can download the file from an originating client.

- i. **[1h]: “subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device”**

216. For the reasons above for claim [1c], and below, Sigurdsson in combination with Shappell and Rao discloses and renders obvious claim [1h], **transferring the copy of the first file to the second client device after first transferring the metadata.** Sigurdsson, Fig. 6, ¶¶ 69-74, 80 (describing downloading content to client device after receiving missing content list (metadata)); Shappell, ¶¶ 62, 79, 81; Rao, ¶ 63 (“The second phase of synchronization is to synchronize the selected raw data to the client system.”);.

- j. **[1i]: “wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration”**

217. As explained above for claim [1c], Sigurdsson in combination with Shappell and Rao discloses and renders obvious claim [1i]. Sigurdsson’s “first client” and/or Server 106 obviously would implement, including in view of Shappell and Rao, the claimed **priority assignment configuration** to assign greater priority to metadata than a priority assigned to files, as explained previously for [1c].

2. Claim 2

218. Claim 2 depends from claim 1 and further recites “**wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.**” The limitations of claim 2 are disclosed by and obvious over the disclosures in Sigurdsson in combination with Shappell and Rao.

219. For the reasons above for claim 1 and the reasons below, Sigurdsson in combination with Rao and Shappell discloses and renders obvious claim 2. As discussed above for [1e]-[1f], Shappell renders obvious modifying Sigurdsson so that, **before the copy of the first file is transferred to the second client device, the transfer of the first metadata causes a file representation (icon) on the user interface of the second device to be updated based on the first metadata.** As discussed for [1e]-[1f], Shappell teaches that the receipt of metadata from another computer’s “modification of a file” “will ***change the visual representation of the file*** to a ‘missing’ file” available for download. (Shappell, ¶81.) All of the obvious GUI design options for presenting the indicator proximate the icon discussed for [1e]-[1f] apply equally here, all of which update the file representation (icon).

3. Claim 3

220. Claim 3 depends from claim 1 and further recites “**wherein the computer program instructions, when executed, cause the server system to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the**

second client device.” The limitations of claim 3 are disclosed by and obvious over the disclosures in Sigurdsson in combination with Shappell.

- b. [3p] and [3a]: **“The system of claim 1, wherein the computer program instructions, when executed, cause the server system to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;”**

Claims [3p] and [3a] are disclosed by and would have been obvious in view of Sigurdsson and Shappell.

221. Sigurdsson’s clients generally “contain similar components and can accept and transmit content in a similar manner.” Sigurdsson, ¶¶ 50-51, 26-28 (describing, in connection with the embodiment in Figure 1, that a user may work on a given file, switching between home and work computers). Because Clients A and C are described as operating similarly, both could post/push files to Server 106 in the same way using the same components. *Id.* ¶ 51 (“In the system 400, content still can be shared among the Client A 102, the Client B 104 and a Client C 406 using the Server 106 in a method similar to that of method 200 of FIG. 2.”); ¶ 32 (“FIG. 2 is a sequence diagram showing an illustrative method 200 for sharing and

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

synchronizing content across multiple client devices according to the implementation shown in FIG. 1.”); ¶ 50 (“For example, both clients can include Files 346, HTML Files 348, an Index 350, an Event Listener 352, a File to Text/HTML Converter 354 and an HTTPS Client/server 356.”); *see also* Fig. 3 (depicting both Client A and Client B as having identical event listeners and HTTPS client/servers). Because Sigurdsson describes its clients as being fungible, and for the reason above for claim [1b], claims [3p] and [3a] would have been obvious in view of Sigurdsson with respect to Client C (“**second client device**”). *Id.* ¶ 36 (“Step 202 through step 210 can be performed to provide temporary storage of content originating *from any client*.”). As one exemplary “**second file**,” Sigurdsson discloses an “office memo on the Client C.” *Id.* ¶ 58.

- c. [3b]: “**determine that the server system is in communication with the first client device associated with the user; and**”

222. Sigurdsson discloses: “*When a client comes online, it can notify the Server 106 that it is online*, and the Server 106 can update its table of clients with the online status.” *Id.* ¶ 61. This notification sent to the server from a client allows the server to “**determine that the server system is in communication with the first client device associated with the user.**” For the reasons above regarding claims [1b] and [3a], Sigurdsson’s clients all operate similarly, and it would have

been obvious to determine that Client A (“**first client device**”), or any other client, is in communication with the server.

223. Sigurdsson discloses, in connection with the embodiment of Figure 6, a “sequence diagram showing an illustrative method for sharing and synchronizing content” using the “server architecture of FIG. 4 when a client that is offline comes online.” *Id.* ¶ 61. “In this case, ***the system 400 automatically detects that a particular client is offline.*** In one implementation, ***determination of whether a client is online or offline can be made by monitoring polling by the client.*** If no request is received from a client, then the client is assumed to be offline. Similarly, the Server 106 can maintain a table of client IDs and whether each client is online or offline. ***When a client comes online, it can notify the Server 106 that it is online,*** and the Server 106 can update its table of clients with the online status.” *Id.* The notification sent to the Server from a client that comes online allows the server to determine that the server is in communication with the first client device associated with the user. As explained above, Sigurdsson’s various “clients” are fungible, so this same functionality could be applied with respect to Clients A, B, or C in Sigurdsson. *See id.* ¶ 51 (“The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from ***any the*** [sic] ***Clients 102, 104 or 406.*** ***The drop-box, or storage, capabilities of the Server***

106 can be used, for example, if a particular client is offline when the first attempt is made to deliver new content to it. Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the Client C 406 can have communicat[ion] with the Server 106 in a manner similar to the method used by the Clients A or B.”). Thus, for the same reasons above, Sigurdsson discloses that its Server may detect whether Clients A, B, or C are online or offline and in communication or out of communication with the Server.

- d. **[3c]: “automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.”**

224. As noted above, Sigurdsson explains that its clients are fungible in that they contain the same components and may implement the same functionality with respect to sending and receiving files from the server. *See, e.g.*, Sigurdsson, ¶¶ 50-51. Thus, it would have been obvious to a POSA that the functionality described above for claim 1 with respect to Clients A and C for the embodiment in Figure 6, further described in paragraphs 72-74 and 79-80, could be performed in the reverse direction. *Id.* ¶¶ 72-74, 79-80. Namely, Client C could send the “**second file**” to the server, which, in turn, could determine that it is in communication with Client A

(after Client A has been offline, for example). Thereafter, the server could send Client A the “list identifying the missing content” referenced in paragraph 73 and, as described in paragraph 79-80, could send missing content (i.e., the “**second file**”) to Client A. *Id.* ¶¶ 73, 79-80. Transferring the second file to Client A would occur **automatically**, with no user intervention. In addition, the process would be “**responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device,**” because the server would need to check to make sure that Client A is online and that it actually has a copy of the second file before attempting to send it. Moreover, given the fungibility of Sigurdsson’s clients, it would have been obvious to receive a file from Client C at the Server and then automatically transfer the file to Client A in view of the embodiment described in connection with Figure 5, which describes a similar data flow between Clients A and B. *Id.* ¶¶ 52-55 (“In step 508, the server stores the content received from the originating client, and the content is propagated to the second client.”). Sigurdsson’s discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home and work computers and synchronizing both computers to make sure they are updated with the most recent updates on each respective version. For example, Sigurdsson expressly references a client’s work

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

computer and home computer as two exemplary “clients,” and describes example use cases such as a user editing a Word document that is synchronized between the user’s home and work computers: “For example, a user may edit a Word document file at his home computer. When the user saves the document, the document’s content is automatically extracted and transmitted to the user’s computer at work. When the user searches for the modified document later at his work computer, the content is presented to the user even though the Word document was not previously stored on the user’s work computer.” *Id.* ¶¶ 26-28 (also describing Server 106 as optional “drop box” “conduit” between client devices). A POSA would have found it obvious that in this type of use case, after the copy of the Word document (which may be called “Version 1”) is transmitted from the user’s home computer to the user’s work computer, the user may continue to edit that document on the work computer while the user is at work. Based on Sigurdsson’s bidirectional file synchronization architecture as illustrated in Figure 4, a copy of the updated, edited file at the work computer (call it “Version 2”) would then be transmitted from the work computer, through the Server, and back to the user’s home computer. However, Sigurdsson does not provide details about what would happen at the user’s home computer in that circumstance where “Version 2” of the file is received at the home computer that previously contained “Version 1” of the same file. A POSA

would have been motivated to look for solutions to replace old versions of files with new ones as described in this limitation. A POSA would have found it obvious that one possibility would be for the updated modified file (“Version 2”) to replace the outdated older version of the file (“Version 1”) on the home computer. A POSA would have found it obvious to replace the older version of the file for reasons including the obvious benefit of preventing copies of outdated versions from endlessly accumulating in storage on the user’s computers. Furthermore, Shappell expressly teaches and motivates this feature with its “automatic replication” option, as discussed further below. Shappell’s “automatic replication” option further renders obvious implementing Sigurdsson’s Server 106 to **automatically transfer the copy of the second file to the first client device to replace an older version of the file responsive to determining that the server is in communication with the first client device and receiving the copy of the second file from the second client device**. Shappell teaches a user-selectable “File Replication” setting where the user can switch between (1) manually downloading files “on an on-demand basis,” and (2) automatically downloading files when available, with an “automatic replication” setting. Shappell, ¶¶ 62, 81. When “automatic replication” is enabled, Shappell teaches automatically replacing an “out-of-date file” with an “updated version” on a receiving computer. *Id.* ¶ 81 (“When a modification of a file contained

in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file***. This will change the visual representation of the file to a ‘missing’ file within the shared space folder when the application setting does not automatically update files, as is shown in step 2010 in FIG. 20. On the other hand, ***if automatic replication of the shared file is enabled, the local machine obtains an updated version of the file***, as is shown in step 2006 in FIG. 20.”), Figs. 19-20.

225. It would have been obvious to implement Shappell’s “automatic replication” feature as an option for each Sigurdsson client device, including Client A (the “**first client device**”), so that Server 106 would **automatically** transfer to Client A the copy of a file modified at Client C (the “**second client device**”), to ensure that the user at Client A would have the most recently updated file. A POSA would have been motivated to incorporate Shappell’s auto-replication to address Sigurdsson’s concern for each client “to keep its locally stored content synchronized with content from the other clients,” and it easily and obviously would be implemented applying Sigurdsson’s teaching that each client can “maintain an open connection” with the server. Sigurdsson, ¶¶ 84-85.

4. Claim 4

226. Claim 4 depends from claim 1 and further recites “**wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.**”

227. For the reasons above for claim 1 and the reasons below, Sigurdsson in combination with Rao and Shappell discloses and renders obvious claim 4. Sigurdsson discloses that Client A “posts” the copy of the file, which Sigurdsson also refers to as a “push,” responsive to the user modifying the content of the file: “The Client A 102 **posts** Client A Content 108 to the Server 106 when content is changed on the Client A 102. . . . In this connection-oriented architecture . . . , each of the clients can **push** new data” Sigurdsson, ¶ 84; *id.*, ¶ 54 (“For example, referring to FIG. 4, the HTTPS Client/server 326 of the Client A 102 can transmit an HTTP post command containing the Client X Content 410 to the Server 106.”). A POSA would also understand that an HTTP POST command discloses a “push request.” It would have been obvious to incorporate this functionality associated with Figure 7 from Sigurdsson into other embodiments in Sigurdsson, such as the embodiment shown in Figure 6 and discussed above, given that they reference the same components (Server 106 and Clients A, B, and C) and given that the clients are

described as being fungible and used for similar purposes, as discussed above. *Id.*

¶¶ 50-51.

5. Independent Claim 8

228. Claim 8, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 8 (method)
1. A system, comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	8. A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and
(ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and	(ii) the graphical availability indication is presented proximate a graphical file representation of the first file on a user interface of the second client device, and
wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and	wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and
subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,	subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device,
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata

associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

229. The only substantive difference between claim 1 and claim 8 is that the latter recites “[a] method” for performing the same steps recited in claim 1. Thus, for the reasons discussed above in connection with claim 1, claim 8 would have been obvious.

6. Claim 9

230. Claim 9 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 2:

Claim 2 (system)	Claim 9 (method)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.	9. The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

231. Claim 9 presents no material difference from claim 2. Claim 9 is thus obvious for the same reasons as claim 2, as discussed above.

7. Claim 10

232. Claim 10 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 10 (method)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	10. The method of claim 8, further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

233. Claim 10 presents no material difference from claim 3. Claim 10 is thus obvious for the same reasons as claim 3, as discussed above.

8. Claim 11

234. Claim 11 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 11 (method)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	11. The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

235. Claim 11 presents no material difference from claim 4. Claim 11 is thus obvious for the same reasons as claim 4, as discussed above.

9. Independent Claim 13

236. Claim 13, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system,**” cause operations, instead of a “**system.**” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 13 (media)
1. A system, comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to: receive, over a network, a copy of a first	13. One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising: receiving, over a network, a copy of a

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	(i) the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and
(ii) the graphical availability indication is presented proximate a file icon	(ii) the availability indication is presented proximate a graphical file

representing the first file on a user interface of the second client device, and	representation of the first file on a user interface of the second client device, and
wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and	wherein the availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and
subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,	subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device,
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

237. Sigurdsson discloses that Server 106 can include “System 900” in Figure 9. Sigurdsson, ¶ 104 (“System 900” depicted in Figure 9 “may be included in either or all of the Client A 102, the Client B 104, and the **Server 106**.”). System 900 includes “a **Memory 920**” and a “**Storage Device 930**.” *Id.* ¶ 105. Memory 920 may be a “non-volatile memory unit” and “[s]torage devices suitable for tangibly embodying computer program instructions and data include all forms of non-

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

volatile memory.” *Id.* ¶¶ 105-110. Thus, this claim would have been obvious over Sigurdsson, Shappell, and Rao for these reasons and those above for claim 1.

238. System 900 further includes “a **Processor 910**, a Memory 920, a **Storage Device 930**, and Input/Output Devices 940. Each of the components 910, 920, 930, and 940 are interconnected using a System Bus 950. **The Processor 910 is capable of processing instructions for execution within the System 900.** In one implementation, the Processor 910 is a single-threaded processor. In another implementation, the Processor 910 is a multi-threaded processor. **The Processor 910 is capable of processing instructions stored in the Memory 920 or on the Storage Device 930** to display graphical information for a user interface on the Input/Output Devices 940.” *Id.* ¶ 105. Sigurdsson further discloses, in connection with System 900: “The apparatus can be implemented in a **computer program product tangibly embodied in an information carrier, e.g., in a machine-readable storage device** or in a propagated signal, **for execution by a programmable processor**; and **method steps can be performed by a programmable processor executing a program of instructions** to perform functions of the described implementations by operating on input data and generating output. The described features can be implemented advantageously in one or more computer programs that are executable on a programmable system including at least one programmable

processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. ***A computer program is a set of instructions that can be used, directly or indirectly, in a computer to perform a certain activity or bring about a certain result.*** *Id.* ¶ 109; see also *id.* ¶ 110 (“Generally, ***a processor will receive instructions*** and data from a read-only memory or a random access memory or both. The essential elements of a computer are ***a processor for executing instructions*** and ***one or more memories for storing instructions and data.***”).

239. Sigurdsson further discloses: “The Memory 920 stores information within the System 900. In one implementation, the Memory 920 is a computer-readable medium. In another implementation, the Memory 920 is a volatile memory unit. ***In another implementation, the Memory 920 is a non-volatile memory unit.***” *Id.* ¶ 106. “***Storage devices suitable for tangibly embodying computer program instructions and data include all forms of non-volatile memory***, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.” *Id.* ¶ 110.

10. Claim 14

240. Claim 14 depends from claim 13 and, for purposes of my analysis, is

not materially different from claim 2:

Claim 2 (system)	Claim 14 (media)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.	14. The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

241. Claim 14 presents no material difference from claim 2. Claim 14 is thus obvious for the same reasons as claim 2, as discussed above.

11. Claim 15

242. Claim 15 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 15 (media)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	15. The media of claim 13, the operations further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the

content of the second file;	content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

243. Claim 15 presents no material difference from claim 3. Claim 15 is thus obvious for the same reasons as claim 3, as discussed above.

12. Claim 16

244. Claim 16 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 16 (media)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	16. The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

245. Claim 16 presents no material difference from claim 4. Claim 16 is

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

thus obvious for the same reasons as claim 4, as discussed above.

**C. '823 Ground 2: Claims 1-4, 8-11, and 13-16 of the '823 Patent
Would Have Been Obvious in View of Sigurdsson in Combination
with Shappell, Rao, and Moromisato.**

1. Independent Claim 1

246. Claim 1 of the '823 patent would have been obvious for the reasons
discuss above in Ground 1. In addition, it would have been obvious to combine
Sigurdsson with Shappell, Rao, and Moromisato and, in view of these references,
claim 1 would have been obvious.

- b. [1e] and [1f]: “wherein, before the copy of the first file
is transferred to the second client device: (i) the
transfer of the first metadata to the second client device
causes a graphical availability indication of the
updated version of the first file to be presented at the
second client device based on the first metadata, and
(ii) the graphical availability indication is presented
proximate a file icon representing the first file on a user
interface of the second client device”**

247. For the reasons discussed above in Ground 1, and the reasons below,
Sigurdsson in combination with Shappell, Rao, and Moromisato discloses that,
before the copy of the first file is transferred to the second client device, there is a
transfer of the first metadata to the second client device, which causes a graphical
availability indication of the updated version of the first file to be presented at the
second client device based on the first metadata, the graphical availability indication

being presented proximate a file icon representing the first file on a user interface of the second client device.

248. As discussed above, a POSA would have found it obvious to consider how to indicate to users the current state of files being synchronized using Sigurdsson systems and methods and would have been familiar with Microsoft Windows as a well-known operating system including graphical user interface components for indicating the state of file transfers and sharing, as disclosed in Moromisato, for example. Thus, a POSA would have looked to references like Moromisato for insight into how to indicate to users the state of files before, during, and after synchronization via graphical user interfaces.

249. The shared spaces in Moromisato each have an associated document share engine 312, 316, 320 and 324, respectively. Each document share engine stores a “file descriptor” for each file in a synchronized folder. “The file descriptor consists of the *file metadata* (such as filename, size, and modification time) and is used to keep track of the file in the collaborative system. During collaborative system operation, *each engine also sends data change requests containing the file descriptors to each collaborator in its respective shared space in order to keep all collaborators synchronized.*” Moromisato at 8:1-8. Moromisato discloses “stub” files that represent files available from another computer but that have not yet been

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

downloaded onto the user's device. *Id.*, 14:47-56 (“In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.***”). Moromisato explains that download settings on the user's device may determine whether to: “(1) Always download new files; (2) Download new files if the file sizes are smaller than a selected number of kilobytes (where the selected number of kilobytes can be entered into textbox 944) and (3) Download new files manually as specified by the user.” *Id.*, 15:6-10. “If the download settings are such that file downloading is deferred to a later time, ***that file is represented in the OS file system as a ‘stub’ file that is very similar to a Windows shortcut (.Ink file) in that it represents a target file with a small file that contains information necessary to find the target file.*** The stub file is represented in the user interface by the same icon as the target file, but it ***may have an overlay of some sort that indicates that it is a stub file*** in a manner similar to the hand overlay shown in FIG. 1.” *Id.*, 15:14-22.

250. In addition, as discussed above, Moromisato describes a pop-up

“Download” menu option, proximately indicating availability to “Download” a file. *Id.*, 15:36-43 (a “menu with appropriate actions would then appear,” including “Download”).

251. A POSA similarly would have been motivated to exploit “[t]he ‘Details’ view in Windows Explorer [that] displays columns of information about each file in the file system” (*Id.*, 17:14-21), as another obvious preexisting, convenient, and user-friendly Windows feature to graphically indicate, **proximate a representation of the “missing” file**, the significant information of which files are available for download—again, illustrated by Moromisato (*see id.*), complementing Shappell’s own suggestion to use “Windows Explorer-style” displays (Shappell, ¶ 42).

252. A POSA also would have considered obvious, as described in Moromisato, that the icon “that represents that file is **provided with an icon overlay**” that “can be similar to **the ‘hand’ overlay 114 (FIG. 1) displayed by Windows XP Pro** file system when a folder is shared over the network.” Moromisato, 17:3-14, 15:14-22 (where synchronized “**file downloading is deferred to a later time**, that file is represented in the OS file system as a ‘stub’ file that is very similar to a Windows shortcut,” with “the same icon as the target file, but it **may have an overlay of some sort**”), Fig. 1. This familiar Windows “hand” overlay for a network-shared resource

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

discloses another obvious **graphical availability indication proximate a file icon** to be presented proximate a ghosted (or alternatively non-ghosted) icon for a “missing” file that is available for download. The classic Windows “shortcut” icon overlay (or a similar adjacent indicator) also would have been another trivially obvious choice to indicate, **proximate the file’s icon**, that the file is not present locally but is available for download. Moromisato, 15:14-22 (where synchronized “*file downloading is deferred to a later time*, that file is represented in the OS file system as a ‘stub’ file that is *very similar to a Windows shortcut*,” with “the same icon as the target file, but it *may have an overlay of some sort*”).

253. Yet another obviously well-suited, familiar Windows feature to supplement or replace Shappell’s “ghosted” icon was “Infotips”—a pop-up graphic “to display extended information” **proximate a file icon** when mousing over it, again naturally applying Shappell’s motivation to use existing Windows GUI features to provide information about remotely-located files. *Id.*, 17:22-29 (illustrating “infotips” in Windows for displaying information about “synchronized files”).

Motivation to Combine Sigurdsson, Moromisato, Rao, and Shappell

254. For the reasons discussed above in connection with claims [1e] and [1f] in Ground 1, a POSA would have been motivated to consider references like

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Moromisato that describe Microsoft Windows user-interface functionality in view of Sigurdsson, Rao, and Shappell, to improve Sigurdsson's systems and methods by presenting a graphical availability indicator proximate a file icon to indicate the status of a file being synchronized.

255. Sigurdsson, Shappell, Rao, and Moromisato are analogous references in the same field as the '823, directed to synchronizing files and associated metadata. All three references are directed to techniques for effectively synchronizing modified versions of files between devices over a network. And all three references are reasonably pertinent to the problems faced by the inventor of the challenged patent for improving upon existing file synchronization systems.

256. The combination would have straightforwardly combined prior art elements according to known techniques to yield a predictable improvement to Sigurdsson's system, improving the Sigurdsson system in the same way that the relevant features enhance Shappell's, Rao's, and Moromisato's disclosed systems, by automatically transmitting metadata as a priority based on file changes to timely notify users that an updated version of a file is available (Shappell), assigning higher priority to that metadata over a lower priority assigned to files (Rao), and using file metadata to display file graphical availability indicators (Moromisato).

257. A POSA also would have been motivated to consider Shappell, Rao,

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

and Moromisato given their originations with leading file-management vendor Microsoft, as Sigurdsson likewise contemplates standard “home” and “work” computers with Microsoft software. Sigurdsson, ¶¶ 27, 42, 45, 58, 60.

258. Furthermore, a POSA would have been motivated to consider additional ways to address Sigurdsson’s stated bandwidth concerns. *Id.*, ¶¶ 51, 58 (with limited bandwidth, “content sharing can fall behind demand”), 60. A POSA would have recognized that Sigurdsson’s system gives rise to a potential gap in synchronicity between versions of a synchronizing “very large document” over a slow network connection, without notifying the user. *Id.*, ¶ 58. Indeed, as Rao cautions, “synchronizing” a very large set of files “may take many minutes or even several hours, depending on the throughput of the communication path . . . among other things.” Rao, ¶ 48. As an illustration, the user of a synchronizing client device (e.g., work computer) in Sigurdsson may work on modifying the large file. When the user subsequently accesses another synchronizing client device (for example, home computer) to continue work on that file, the user would benefit from being able to quickly and visually determine whether or not the version of the file on the home computer is current, before opening the file to work on it. In view of this potential deficiency in Sigurdsson, a POSA would have been motivated to consider Shappell and Rao for a solution that immediately transmits metadata generated by

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

modifying the file to the user's other client devices (including through Server 106, using its priority algorithm, to account for other client devices that may be currently offline), and would have been further motivated to advantageously leverage the metadata for "any purpose" such as to "**inform a user**" that an updated file is available, as discussed in Moromisato, for example. Shappell, ¶ 79 (motivating use of metadata "[t]o **inform a user** of a group shared space file that is not present locally" by displaying an indicator "**to denote a 'missing' file that is available on another system**"); Rao, ¶¶ 63 (motivating that "**the client system . . . may use and manipulate the metadata for any purpose whatsoever**"), 43 (motivating that the metadata enables that "**certain functions and operations may be available to the user**" of the receiving client, including to "display" information regarding files).

259. A POSA would have reasonably expected success in implementing this modified Sigurdsson system incorporating Moromisato's graphical availability indicator functionality. A POSA would not have been dissuaded from combining Moromisato with Sigurdsson, Rao, and Shappell in view of Shappell's primary embodiment being peer-to-peer rather than client-server. A POSA would have understood that the relevant features of Shappell relating to transfer of metadata and graphical indication of file availability are independent of the type of network architecture as between peer-to-peer or client-server. While Shappell's exemplary

embodiment is “server-less” (e.g., Shappell, Abstract, ¶ 7), Shappell does not specifically disparage a client/server implementation, and proclaims that the “type of network . . . is not critical” and is only “preferably” peer-to-peer, and suggests also transferring “indirectly” between user machines such as would be provided by a client/server architecture (*Id.*, ¶ 40). Similarly, a POSA would not have been dissuaded by Shappell’s statement that, in its exemplary Figure 2 embodiment, “[e]ach user device will **typically** be used by or associated with a different user” (*Id.*), given the compelling applicability of Shappell’s relevant teaching to improve Sigurdsson’s system as explained above.

260. A POSA would have been further motivated to look to references like Moromisato, given that it was assigned to Microsoft Corporation, in view of the popularity of Microsoft Windows at the time and a POSA’s motivation to study the various file sharing and synchronization solutions invented by Microsoft that are disclosed in, e.g., Shappell (which indicates on its cover page that it was assigned to Microsoft) and Moromisato (same). As noted above, a POSA would have known that Microsoft was a significant vendor at the time in file management solutions, including file sharing solutions such as those described in Shappell.

261. A POSA would have had at least a reasonable expectation of success in implementing the modified system of Sigurdsson in this manner. Sigurdsson already

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

discloses the presence of metadata and the use of standard personal computers and operating systems that were very familiar to a POSA. Transmitting metadata and providing a graphical availability indication as taught by Shappell and Moromisato would have been well within the grasp of a POSA, as well as implementing the modified system of Sigurdsson accordingly, using well-known computer software and networking knowledge and design principles. The references provide ample detail such that a POSA would have been able to implement the modification without undue experimentation.

- c. **[1g]: “wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device”**

262. For the reasons discussed above in Ground 1 and the prior elements of claim 1, and the reasons below, Sigurdsson in combination with Shappell, Rao, and Moromisato discloses a graphical availability indication that indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device.

263. For the reasons discussed above in Ground 1 in connection with claims [1e] and [1f] and the other foregoing limitations of claim 1 in this Ground,

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Sigurdsson in combination with Shappell and Rao discloses a graphical user interface and functionality for sending updated files from one client to a server and, in turn, from the server to another client. In addition, it would have been obvious to look to references like Shappell and Moromisato for implementations of user interface indications that updated versions of files are available for download.

264. The shared spaces in Moromisato each have an associated document share engine 312, 316, 320 and 324, respectively. Each document share engine stores a “file descriptor” for each file in a synchronized folder. “The file descriptor consists of the ***file metadata*** (such as filename, size, and modification time) and is used to keep track of the file in the collaborative system. During collaborative system operation, ***each engine also sends data change requests containing the file descriptors to each collaborator in its respective shared space in order to keep all collaborators synchronized.***” Moromisato at 8:1-8. Moromisato discloses “stub” files that represent files available from another computer but that have not yet been downloaded onto the user’s device. *Id.*, 14:47-56 (“In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.”).

Moromisato explains that download settings on the user’s device may determine whether to: “(1) Always download new files; (2) Download new files if the file sizes are smaller than a selected number of kilobytes (where the selected number of kilobytes can be entered into textbox 944) and (3) Download new files manually as specified by the user.” *Id.*, 15:6-10. “If the download settings are such that file downloading is deferred to a later time, ***that file is represented in the OS file system as a ‘stub’ file that is very similar to a Windows shortcut (.Ink file) in that it represents a target file with a small file that contains information necessary to find the target file.*** The stub file is represented in the user interface by the same icon as the target file, but it ***may have an overlay of some sort that indicates that it is a stub file*** in a manner similar to the hand overlay shown in FIG. 1.” *Id.*, 15:14-22. It would have been obvious to practice this claim limitation in view of Sigurdsson in combination with Moromisato by, for example, using the graphical availability indication of Moromisato to indicate that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server disclosed in Sigurdsson to the second client device.

265. Moromisato’s graphical availability indication indicates that the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

updated version of the file is available to be downloaded from the server system.

Id., 15:36-38 (“In one embodiment, ***the user can download the target file associated with a stub file by double clicking on the stub file icon.***”), *id.*, 16:27-51 (“***[T]he user/invitee at some point will download the file either by double clicking on it or by selecting the file in a list and then using the file tasks pane to download the file.***”).

The Windows shell extension 306 is invoked as discussed above and accesses the collaborative system via the web services interface. It uses the files tool in the shared space to fetch the file. When a file is being downloaded, a download pane 1370 such as that illustrated in FIG. 13 appears in the synchronize pane 1320. The download pane 1370 displays the status of the download, for example, by means of a progress bar 1372 or other means. A “Cancel” button 1374 allows the user to cancel the download. The download pane 1370 always applies to the selected file. If there is no selection but some files are being downloaded then, in one embodiment, the download pane may show a statement such as “Downloading 3 files” in place of the progress bar 1372. If there are multiple files selected and all the files are stubs being downloaded then a similar statement can be shown. The information in this pane is provided by the Windows shell extension that, in turn, uses the files tool in the shared space (via the web services interface) to get the required information. ***When the file arrives, the stub file representing the actual file is replaced by the actual file*** and

the file system snapshot is updated to indicate that the file contents have changed.”).

d. [1h]: “subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device”

266. For the reasons discussed above in Ground 1 and the prior elements of claim 1, and the reasons below, Sigurdsson in combination with Shappell, Rao, and Moromisato discloses a transfer of the copy of the first file to the second client device subsequent to the transfer of the first metadata to the second client device.

267. Moromisato discloses that its stub files, which (as discussed above) only appear after receiving metadata associated with files, may be replaced when copies of the files are downloaded. Moromisato at 15:36-38 (“In one embodiment, ***the user can download the target file associated with a stub file by double clicking on the stub file icon.***”), *id.*, 16:27-51 (“***[T]he user/invitee at some point will download the file either by double clicking on it or by selecting the file in a list and then using the file tasks pane to download the file.*** The Windows shell extension 306 is invoked as discussed above and accesses the collaborative system via the web services interface. It uses the files tool in the shared space to fetch the file. When a file is being downloaded, a download pane 1370 such as that illustrated in FIG. 13 appears in the synchronize pane 1320. The download pane 1370 displays the status of the download, for example, by means of a progress bar 1372 or other means. A

“Cancel” button 1374 allows the user to cancel the download. The download pane 1370 always applies to the selected file. If there is no selection but some files are being downloaded then, in one embodiment, the download pane may show a statement such as “Downloading 3 files” in place of the progress bar 1372. If there are multiple files selected and all the files are stubs being downloaded then a similar statement can be shown. The information in this pane is provided by the Windows shell extension that, in turn, uses the files tool in the shared space (via the web services interface) to get the required information. ***When the file arrives, the stub file representing the actual file is replaced by the actual file*** and the file system snapshot is updated to indicate that the file contents have changed.”). It would have been obvious in view of these disclosures in Moromisato to use Sigurdsson’s server to transfer the copy of the first file to the second client device subsequent to the transfer of the first metadata to the second client device.

2. Claim 2

268. Claim 2 depends from claim 1 and further recites “**wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.**” For the reasons discussed above in Ground 1 and in

connection with claim 1 for this Ground, and the reasons below, the limitations of claim 2 are disclosed by and obvious over the disclosures in Sigurdsson in combination with Shappell and Moromisato.

269. A POSA would have found it obvious to consider how to indicate to users the current state of files being synchronized using Sigurdsson systems and methods and would have been familiar with Microsoft Windows as a well-known operating system including graphical user interface components for indicating the state of file transfers and sharing, as disclosed in Shappell and Moromisato, as described above. Thus, a POSA would have looked to references like Shappell and Moromisato for insight into how to indicate to users the state of files before, during, and after synchronization via graphical user interfaces.

270. The shared spaces in Moromisato each have an associated document share engine 312, 316, 320 and 324, respectively. Each document share engine stores a “file descriptor” for each file in a synchronized folder. “The file descriptor consists of the ***file metadata*** (such as filename, size, and modification time) and is used to keep track of the file in the collaborative system. During collaborative system operation, ***each engine also sends data change requests containing the file descriptors to each collaborator in its respective shared space in order to keep all collaborators synchronized.***” Moromisato at 8:1-8. Moromisato further discloses:

“Each document share engine also holds the aforementioned file system snapshot that the file synchronizer needs to ***detect file changes***. Depending on how changed files are distributed, a document share engine may also keep a copy of the contents of a synchronized file in the engine so that when the file is edited (as determined by the file synchronizer), the copy in the document share engine can be used as a point of comparison.” *Id.*, 8:8-15.

271. Moromisato discloses “stub” file representations, and it would have been obvious to use them to change file representations from regular file representations to indications that the file has been updated on another user device but has not yet been downloaded onto the user’s device. *Id.*, 14:47-56 (“In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.***”). Moromisato explains that download settings on the user’s device may determine whether to: “(1) Always download new files; (2) Download new files if the file sizes are smaller than a selected number of kilobytes (where the selected

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

number of kilobytes can be entered into textbox 944) and (3) Download new files manually as specified by the user.” *Id.*, 15:6-10. “If the download settings are such that file downloading is deferred to a later time, ***that file is represented in the OS file system as a ‘stub’ file that is very similar to a Windows shortcut (.Ink file) in that it represents a target file with a small file that contains information necessary to find the target file.*** The stub file is represented in the user interface by the same icon as the target file, but it ***may have an overlay of some sort that indicates that it is a stub file*** in a manner similar to the hand overlay shown in FIG. 1.” *Id.*, 15:14-22.

272. It would have been obvious in view of these disclosures from Moromisato to cause a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata in the systems and methods of Sigurdsson. For example, Client A could send first metadata associated with a file to Sigurdsson’s Server, which, in turn, could send the first metadata to Client C when it comes back online, as discussed above. In so doing, Client C could update its display of the file that it now knows has been updated to show something like the stub file representations described in Moromisato to indicate to the user that the file has been updated but that the actual file content has not yet been downloaded to the client.

3. Claim 3

273. Claim 3 depends from claim 1 and further recites “**wherein the computer program instructions, when executed, cause the server system to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.**” For the reasons discussed above in Ground 1 and in connection with claim 1 for this Ground, and the reasons below, the limitations of claim 3 are disclosed by and obvious over the disclosures in Sigurdsson in combination with Shappell and Moromisato.

- b. [3c]: “automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.”

274. It would have been obvious to consider Moromisato’s teachings about automatically transferring files, in view of Moromisato being an analogous reference and in view of the additional motivations discussed above for a POSA to consider Moromisato. Moromisato discloses a server-based system in which a change to the server data automatically causes the server to send a command to all clients to modify their data: “Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.” Moromisato, 1:47-54, 14:47-55.

275. Moromisato also teaches the automatic transfer of the modified file copy to another device to replace an older version of the file stored on the other device with the modified copy that includes the content modified by the user. For example, Moromisato teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system “sites”. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may either disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

276. It would have been obvious to incorporate this functionality into Sigurdsson’s system in view of the motivation to keep a user’s work and home

computers (as well as the computers of other team members, as taught in Moromisato) synchronized automatically as changes are made to avoid the situation in which people view/edit files that are not up-to-date.

277. A POSA would have been motivated to look to references like Moromisato, which also discloses “synchronizer” functionality—specifically, functionality designed to operating in conjunction with the Windows operating system.¹² Moromisato at 7:9-24 (“The file system RAMP program 304 is part of the collaborative system code and detects changes to files in the OS file system 302. It communicates with the **file synchronizer 308**, as shown schematically by arrow 340, in order to inform the file synchronizer 308 that it needs to synchronize those changes. In particular, ***the Windows XP Pro operating system provides a system event notification system to which clients can subscribe to receive notifications of selected events. These events include changes in the OS file system, such as folder and file edits***, additions and deletions. . . . During the collaborative system start up, the file system RAMP program 304 is invoked so that it can register with the OS

¹² Moromisato, assigned to Microsoft, describes functionality related to the then- and now-popular Microsoft Windows operating system. A POSA—and almost everyone else using computers at the time—would have been familiar with the basic file system concepts described in Moromisato, and a POSA would have been motivated to look at Microsoft patents and patent publications at the time as an industry leader in this technical field.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

event notification system for file notifications indicating **changes to files** and file contents in the Windows shell file system 302.”).

278. Like Sigurdsson, Moromisato describes embodiments that use a “server based” system to implement file synchronization functionality. *Id.*, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. **Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.**”). A POSA considering Sigurdsson’s similar server-based embodiments would have been motivated to consider the design of Moromisato’s file synchronizer illustrated in Figure 3:

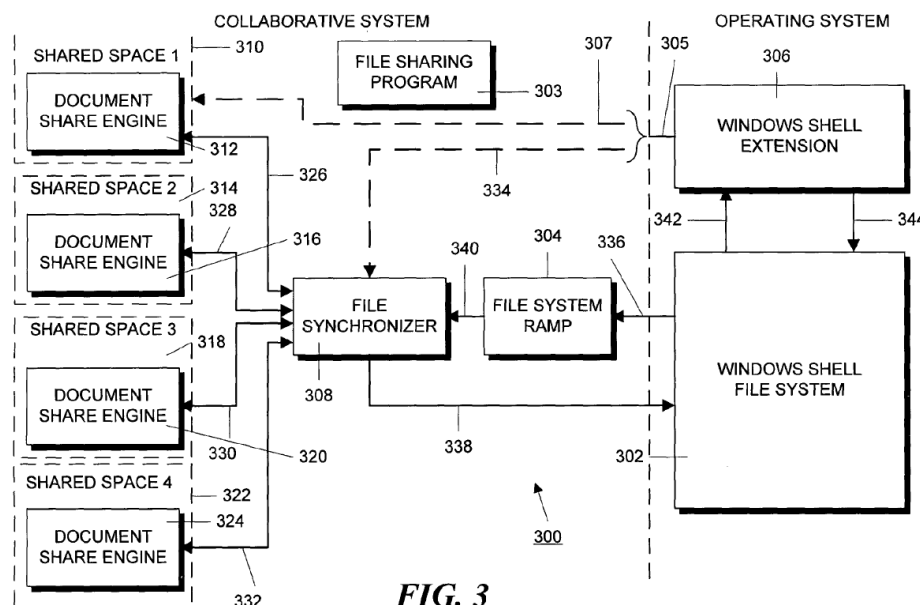


FIG. 3

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Id., Fig. 3; *see also id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system and vice versa.”).

279. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified first electronic file copy to a third electronic device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). *Id.*, 20:8-16; *id.*, 6:42-44 (“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer **replaces** an older version of the file stored on another device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***.”); *see also id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing).

280. In view of these teachings from Moromisato, it would have been

obvious to use the Synchronization Manager in Sigurdsson's system to automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device [for the reasons discussed above in connection with claim element [3b] in Ground 1] and (ii) receiving the copy of the second file from the second client device.

4. Claims 4, 8-11, and 13-16

281. For the reasons discussed above for claim 1, and the reasons discussed above in the corresponding sections of Ground 1, claims 4, 8-11, and 13-16 would have been obvious in view of Sigurdsson in combination with Shappell, Rao, and Moromisato.

D. '787 Ground 1: Claims 1-4, 8-11, and 13-16 of the '787 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Rao.

1. Independent Claim 1

282. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

[1d] automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[1e] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and

[1f] (ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

[1g] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

283. As shown in the table below, claim 1 of the '787 patent is similar to claim 1 of the '823 patent (substantively identical limitations appear on the same row).

'823 Claim 1	'787 Claim 1
1. A system, comprising:	1. A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of	automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

the first file is transferred to the second client device,	the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	
	(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
(ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and	
	(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and
wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and	

subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,	
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

284. The limitations that are substantively identical between claim 1 of the '823 patent and claim 1 of the '787 patent would have been obvious for the reasons discussed above in Ground 1 for the '823 patent.

- b. [1e]: “wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and”**

285. Sigurdsson in combination with Shappell discloses that, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the

first metadata.

286. For the reasons discussed above in connection with '823 claims [1e] and [1f], it would have been obvious to transfer the first metadata before the copy of the first file is transferred to the second client device, and it would have been obvious to look to references like Shappell in view of Sigurdsson's brief reference to user interfaces to find more detail about potential user interface features that could be used to indicate to users the state of files before, during, and after synchronization.

287. In addition, it would have been obvious in view of Sigurdsson in combination with Shappell to **transfer the first metadata to the second client device to cause a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata.**

288. Shappell discloses ways to *"inform a user of a group shared space file that is not present locally"* such that *"display icons are presented to the user."*

Shappell, ¶ 79. Shappell describes:

FIG. 15 illustrates a graphical interface 1500 for a Shared Space folder that contains files that are not local. As shown, ***a left hand pane 1502 presents a display icon 1504 to denote a 'missing' file that is available on another system. This icon is ghosted or semi-transparent.*** As is shown in the flow chart in FIG. 20, ***the system renders such a display upon receipt that metadata concerning a shared file is available in the peer network,*** as is shown in method step 2002. Other files that are 'missing' may be unavailable to the user, such as when ***a system that is hosting the file of interest is offline.*** In this instance, a display icon 1506 is presented as a ghosted icon that includes ***a small image of***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

a red “X” or other suitable visual indicia that the file is unavailable to the user.

Id. This excerpt describes at least two examples of **updates to file representations**.

First, an “icon” representing a missing file may be “ghosted or semi-transparent” to indicate that an updated file is “available on another system.” *Id.* Second, the “small image of a red ‘X’” on a file’s icon indicates that the file is available on another “system that is hosting the file of interest” but is currently “offline.” *Id.*

289. Shappell further discloses that these file representations are updated **“upon receipt” of “metadata concerning a shared file” that “is available.”** Shappell, ¶ 79; *see also id.*, ¶ 81 (“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will remove the local copy of the out-of-date file. ***This will change the visual representation of the file to a “missing” file within the shared space folder*** when the application setting does not automatically update files, as is shown in step 2010 in FIG. 20. On the other hand, if automatic replication of the shared file is enabled, the local machine obtains an updated version of the file, as is shown in step 2006 in FIG. 20. Finally, an appropriate visual indicator is presented to the user as shown in step 2008.”); *see also id.*, ¶ 76 (“For sharing files among group members, ***metadata***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

is transmitted to all members when files become available, such as is shown in the steps 1904 and 1906 in the flow chart in FIG. 19. ***Such metadata are sufficient to render an icon of the shared file*** and may include date, time, thumb-nail information, name size, and optionally the source of the information.”). For the reasons above and in connection with ’823 claims [1e] and [1f], it would have been obvious in view of these disclosures for **the transfer of the first metadata to the second client device to cause a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata.**

290. The same motivations to combine Sigurdsson with Shappell and Rao described above for ’823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Shappell into Sigurdsson’s systems and methods to practice claim [1e].

- c. [1f]: **“(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and”**

291. Sigurdsson in combination with Shappell discloses that instead of the updated file representation of the first file representing a version of the first file

currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device.

292. For the reasons discussed above in connection with claim [1e], Sigurdsson discloses a graphical user interface, and it would have been obvious to look to references like Shappell for implementations of user interface file representations related to file synchronization functionality.

293. As discussed above for claim [1e], Shappell describes at least two examples of **updates to file representations**. Shappell, ¶ 79. First, an “icon” representing a missing file may be “ghosted or semi-transparent” to indicate that an updated file is “available on another system.” *Id.* Second, the “small image of a red ‘X’” on a file’s icon indicates that the file is available on another “system that is hosting the file of interest” but is currently “offline.” *Id.* Both of these examples describe that, **instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device.** In particular, both exemplary updated file representations are expressly described in Shappell as representing versions of files

“available on another system” or available on another “system that is hosting the file,” respectively. *Id.* Shappell further discloses that, upon learning of an updated version of a file, Shappell’s clients may “remove the local copy of the out-of-date file.” *Id.*, ¶ 81. Thus, the updated file representation would represent a version of the file that is stored on another client (e.g., the originating “system that is hosting the file of interest”), and it would have been obvious for the file representation to represent the updated version of the first file that is currently stored on the first (originating) client device and not currently stored on the second (receiving) client device.

294. For example, as discussed above in connection with ’823 Ground 1 (e.g., for claims [1c] and 3), Sigurdsson’s Client A (**first client device**) is described as being a user’s home computer and Client C (**second client device**) is described as being a user’s mobile device, and it would have been obvious to attempt to keep files stored on both devices synchronized and up-to-date by pushing changes from one client to the other. Thus, it would have been obvious to indicate on a user interface on Client C that the most recent version of a file is currently stored on Client A (the device where the file was modified) upon receipt of **first metadata**, particularly in view of Sigurdsson’s embodiment in paragraphs 72-74 describing Client C learning of missing files via such metadata—and such metadata may

include identification of the originating client (Client A) where the file is stored.

295. For the reasons above and in connection with '823 claim [1e] and [1f], it would have been obvious to incorporate the abovementioned functionality from Shappell into the systems and methods of Sigurdsson to include a change of visual representation of a file to indicate that the file has recently been modified when a metadata-based notification of a modification of a file contained in a shared space occurs to indicate the file is available on another client—for example, the server in Sigurdsson could send metadata to Client C to indicate that an updated version of a file is available on Client A prior to actually sending the file, and Shappell's user interface functionality could be used to properly notify the user of this situation, as discussed above.

296. The same motivations to combine Sigurdsson with Shappell and Rao described above for '823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Shappell into Sigurdsson's systems and methods to practice claim [1f].

2. Claim 2

297. Claim 2 depends from claim 1 and further recites “**wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability**

indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.” Aside from the reference to the “second client device” having a “user interface,” this limitation is substantively identical to ’823 element [1e] and would have been obvious in view of Sigurdsson and Shappell for the reasons discussed above in connection with ’823 element [1e] and the reasons below.

298. Sigurdsson discloses that its server and any of its clients may include graphical user interfaces: “For example, the *System 900 may be included in either or all of the Client A 102, the Client B 104*, and the Server 106.” Sigurdsson, ¶ 104. As noted above in connection with ’823 claim 3, user-interface functionality could be implemented by any of the clients described in Sigurdsson, given that Sigurdsson describes its clients as being fungible. *Id.*, ¶¶ 50, 108 (“The Input/Output Devices 940 provides input/output operations for the System 900. In one implementation, the Input/Output Devices 940 includes a keyboard and/or pointing device. In another implementation, the Input/Output Devices 940 include a display unit for displaying graphical user interfaces.”); *see also id.*, ¶ 111 (“To provide for interaction with a user, the features can be implemented on a computer having a display device such as a CRT (cathode ray tube) or LCD (liquid crystal display) monitor for displaying information to the user and a keyboard and a pointing device

such as a mouse or a trackball by which the user can provide input to the computer.”).

3. Claim 3

299. Claim 3 depends from claim 1 and further recites “**wherein the computer program instructions, when executed, cause the server system to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.**” The limitations of ’787 claim 3 are substantively identical to those of ’823 claim 3 and would have been obvious for the reasons discussed above in connection with ’823 claim 3.

4. Claim 4

300. Claim 4 depends from claim 1 and further recites “**wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.**” The limitations of ’787 claim 4 are substantively identical to those of ’823 claim 4 and would have been obvious for the reasons discussed above in connection with ’823 claim 4.

5. Independent Claim 8

301. Claim 8, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system.**” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 8 (method)
A system comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the	receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

user modifying the content of the first file;	from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and	receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and	(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and	(ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and
wherein at least one of the server system	wherein at least one of the server system

or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

302. The only substantive difference between claim 1 and claim 8 is that the latter recites “[a] method” for performing the same steps recited in claim 1. Thus, for the reasons discussed above in connection with claim 1, claim 8 would have been obvious.

6. Claim 9

303. Claim 9 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 2:

Claim 2 (system)	Claim 9 (method)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.	9. The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

304. Claim 9 presents no material difference from claim 2. Claim 9 is thus obvious for the same reasons as claim 2, as discussed above.

7. Claim 10

305. Claim 10 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 10 (method)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	10. The method of claim 8, further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

306. Claim 10 presents no material difference from claim 3. Claim 10 is thus obvious for the same reasons as claim 3, as discussed above.

8. Claim 11

307. Claim 11 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 11 (method)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	11. The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

308. Claim 11 presents no material difference from claim 4. Claim 11 is thus obvious for the same reasons as claim 4, as discussed above.

9. Independent Claim 13

309. Claim 13, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system,**” cause operations, instead of a “**system.**” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Claim 1 (system)	Claim 13 (media)
A system comprising:	One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and	receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to	(i) the transfer of the first metadata to

the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and	the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and	(ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

310. The limitation reciting a “**non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system,**” cause operations would have been obvious for the reasons discussed above in connection with claim 13 of the ’823. Thus, for the reasons discussed above for claims 1 and 13 of the ’823, this claim would have been obvious.

10. Claim 14

311. Claim 14 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 2:

Claim 2 (system)	Claim 14 (media)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.	14. The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

312. Claim 14 presents no material difference from claim 2. Claim 14 is thus obvious for the same reasons as claim 2, as discussed above.

11. Claim 15

313. Claim 15 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 15 (media)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	15. The media of claim 13, the operations further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second

file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

314. Claim 15 presents no material difference from claim 3. Claim 15 is thus obvious for the same reasons as claim 3, as discussed above.

12. Claim 16

315. Claim 16 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 16 (media)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	16. The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

316. Claim 16 presents no material difference from claim 4. Claim 16 is thus obvious for the same reasons as claim 4, as discussed above.

E. '787 Ground 2: Claims 1-4, 8-11, and 13-16 of the '787 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell, Rao, and Moromisato.

1. Independent Claim 1

317. Claim 1 of the '787 patent would have been obvious for the reasons discuss above in Ground 1. In addition, it would have been obvious to combine Sigurdsson with Shappell, Rao, and Moromisato and, in view of these references, claim 1 would have been obvious.

- b. [1e]: “wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata”**

318. For the reasons discussed above in Ground 1, and the reasons below, Sigurdsson in combination with Shappell, Rao, and Moromisato discloses that, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata.

319. As discussed above, a POSA would have found it obvious to consider how to indicate to users the current state of files being synchronized using Sigurdsson systems and methods and would have been familiar with Microsoft Windows as a well-known operating system including graphical user interface components for indicating the state of file transfers and sharing, as disclosed in Moromisato, for example. Thus, a POSA would have looked to references like Moromisato for insight into how to indicate to users the state of files before, during, and after synchronization via graphical user interfaces.

320. As discussed above for '823 Ground 2 (claims [1e] and [1f]), Moromisato discloses stub files that appear as updated file representations based on receipt of **first metadata** to indicate that the updated files are not present on a client device but are available for download from another source. Moromisato, 14:47-56. Moromisato discloses that a stub file “***represents a target file with a small file that contains information necessary to find the target file.***” *Id.*, 15:14-22 (where synchronized “***file downloading is deferred to a later time***, that file is represented in the OS file system as a ‘stub’ file that is very similar to a Windows shortcut[,]” with “the same icon as the target file, but it ***may have an overlay of some sort***”).

321. In addition, as discussed above in '823 Ground 2, Moromisato describes a pop-up “Download” menu option, Moromisato, 15:36-43; “[t]he

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

‘Details’ view in Windows Explorer [that] displays columns of information about each file in the file system” *id.*, 17:14-21; and “Infotips”—a pop-up graphic “to display extended information” about remotely-located files, *id.*, 17:22-29. Each of these is an additional example of a **file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata**. For example, when the file is updated, the “Download,” “Details,” and “Infotips” features reflect that the updated file is available from a remote source. Thus, it would have been obvious to incorporate these exemplary file representations into Sigurdsson’s systems and methods to indicate file availability for download from a remote source.

322. In addition, for the reasons provided above in Ground 1, it would have been obvious to use **the transfer of the first metadata to the second client device to cause one or more of these file representations of the first file to update their presentation on a user interface of the second client device**. For example, it would have been obvious to use the “file descriptor” metadata, as described above in connection with ’823 Ground 2, for such purpose.

323. The same motivations to combine Sigurdsson with Shappell, Rao, and Moromisato described above for ’823 Ground 2 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from

Moromisato into Sigurdsson's systems and methods to practice claim [1e].

- c. **[1f]: “(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device”**

324. For the reasons discussed above in Ground 1, and the reasons below, Sigurdsson in combination with Shappell and Moromisato discloses that, instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device.

325. As discussed above in connection with '823 claim [1e] and [1f], Moromisato discloses using an “overlay of some sort that indicates that” a file is a “stub file,” which, in turn, indicates that the file is not present on the client but is available for download from another device. Moromisato, 15:14-22. For the reasons above for claim [1e], it would have been obvious in view of these disclosures from Moromisato to cause a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata in the systems and methods of Sigurdsson. For example, Client A could send first

metadata associated with a file to Sigurdsson's server, which, in turn, could send the first metadata to Client C when it comes back online, as discussed above. In so doing, Client C could update its display of the file that it now knows has been updated to show something like the stub file representations described in Moromisato to indicate to the user that the file has been updated but that the actual file content has not yet been downloaded to the client.

2. Claim 2

326. Claim 2 depends from claim 1 and further recites “**wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.**” This claim would have been obvious for the reasons discussed above in Ground 1, the reasons above for claim 1, and the reasons above for claim 2 of Ground 2 for the '823.

3. Claim 3

327. Claim 3 depends from claim 1 and further recites “**wherein the computer program instructions, when executed, cause the server system to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the**

second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.” For the reasons discussed above in Ground 1, and the reasons above for claim 2 of Ground 2 for the ’823, claim 3 would have been obvious.

4. Claims 4, 8-11, and 13-16

328. For the reasons discussed above for claim 1, and the reasons discussed above in the corresponding sections of Ground 1, claims 4, 8-11, and 13-16 would have been obvious in view of Sigurdsson in combination with Shappell, Rao, and Moromisato.

F. '607 Ground 1: Claims 1, 3-5, 7, 12, 14, 17, 19, and 20 of the '607 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Rao.

1. Independent Claim 1

329. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1d] determine that the server system is not in communication with a second client device associated with the user;

[1e] store the copy of the first file on the server system;

[1f] automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device;
and

[1g] automatically transfer, over a network, the copy of the first file to

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

330. As shown in the table below, claim 1 of the '607 patent is similar to claim 1 of the '823 patent (substantively identical limitations appear on the same row).

'823 Claim 1	'607 Claim 1
1. A system, comprising:	1. A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
	determine that the server system is not in communication with a second client

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

	device associated with the user;
	store the copy of the first file on the server system;
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
	automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.
wherein, before the copy of the first file is transferred to the second client device:	
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	
(ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and	
wherein the graphical availability indication indicates that the updated version of the first file generated from	

the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and	
subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,	
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	

331. The limitations that are substantively identical between claim 1 of the '823 patent and claim 1 of the '607 patent would have been obvious for the reasons discussed above in Ground 1 for the '823 patent.

b. [1d]: “determine that the server system is not in communication with a second client device associated with the user;”

332. Sigurdsson discloses determining whether the server system is not in communication with a second client device associated with the user.

333. Sigurdsson discloses, in connection with the embodiment of Figure 6, a “sequence diagram showing an illustrative method for sharing and synchronizing

content” using the “server architecture of FIG. 4 when a client that is offline comes online.” Sigurdsson, ¶ 61. “In this case, ***the system 400 automatically detects that a particular client is offline.*** In one implementation, ***determination of whether a client is online or offline*** can be made by monitoring polling by the client. If no request is received from a client, then ***the client is assumed to be offline.*** Similarly, the Server 106 can maintain a table of client IDs and whether each client is online or offline. When a client comes online, it can notify the Server 106 that it is online, and the Server 106 can update its table of clients with the online status.” *Id.*

334. For example, in connection with the embodiment in Figure 6, Sigurdsson discloses determining whether “Client C” is offline (which would, of course, mean that Client C would not be in communication with the Server). Sigurdsson, ¶ 66. As explained above, Sigurdsson’s various “clients” are fungible, so the same functionality could be applied with respect to Clients A, B, or C in Sigurdsson. *See, e.g., id.*, ¶ 51 (“The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from ***any the*** [sic] ***Clients 102, 104 or 406.*** ***The drop-box, or storage, capabilities of the Server 106 can be used, for example, if a particular client is offline when the first attempt is made to deliver new content to it.*** Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the Client C 406 can have

communicate [sic] with the Server 106 in a manner similar to the method used by the Clients A or B.”).

335. In view of the above, and for the reasons discussed above for ’823 Ground 1 (claims [1b] and [3a]), Sigurdsson’s clients all operate similarly, and it would have been obvious to determine that Client C (“**second client device**”), or any other client, is **not in communication with the server**.

c. [1e]: “store the copy of the first file on the server system;”

336. Sigurdsson discloses storing the copy of the first file on the server system.

337. Sigurdsson discloses that its Server may store the modified **first file** to a memory device in the Server. Sigurdsson, ¶ 48 (“The **Server 106 can contain a Storage 330**, a Synchronization Manager 332, and an Authentication Manager 334. The **Storage 330 can contain the Content A, and Content B through Content N 336 that has been received from the user’s various clients.**”); *id.*, ¶ 74 (“the content may be stored at the server”). Sigurdsson further discloses: “The Server 106 serves as a **temporary storage location** that can facilitate synchronization between the user’s clients.” *Id.*, ¶ 83; *see also id.*, ¶ 85 (“The Client A Content 108 is received by and **stored in the Server 106.**”).

338. Server 106 further includes “a Space Quota 342 and a Throughput

Threshold 344” that are used to control the storage of data to the server. *Id.*, ¶ 48.

339. Sigurdsson also discloses “System 900” in Figure 9 and explains that **“the System 900 may be included in** either or all of the Client A 102, the Client B 104, and **the Server 106.”** Sigurdsson, ¶ 104. In connection with the embodiment in Figure 9, Sigurdsson discloses the use of a memory device: “Storage devices suitable for tangibly embodying computer program instructions and data include ***all forms of non-volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.***” *Id.*, ¶ 110.

- d. [1g]: “**automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.**”

340. Sigurdsson in combination with Shappell discloses automatically transferring, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

341. For the reasons discussed above in '823 Ground 1, claim [1b], all of Sigurdsson's clients communicate with Server 106 **over a network**.

342. For the reasons discussed above in '823 Ground 1, claim [3c], Sigurdsson discloses **automatically transferring the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device**. Although the discussion above for '823 Ground 1, claim [3c], relates to automatically transferring a file from Client C (**second client device**) to Client A (**first client device**), for the reasons discussed above for '823 Ground 1, claim [3c], it would have been obvious to perform the operations recited in '823 claim [3c] in the reverse direction to transfer the file from Client A to Client C. Sigurdsson discloses doing so **responsive to (i) resuming communication with the second client device**.

343. For example, in connection with the embodiment depicted in Figure 6, Sigurdsson describes an automated process for (1) detecting when Client C comes back online, (2) passing a list containing metadata to enable Client C to see what content it is missing, (3) receiving a request from Client C for missing data, and (4) sending the missing data to Client C. The transfer of "Client X Content 410"

could be sent automatically to a client that was temporarily offline when it comes back online. Sigurdsson, ¶ 69 (“Step 612 can occur *after the third client comes online, having been offline during the time that the content originated on the first client and was propagated to the second client and to the server*. In step 612, the third client requests a list of missing content from the server. For example, *the Client C 406 requests a list of missing content from the Server 106 when it connects to network that links the server and other clients, such as the Internet.*”). “In step 614, the server determines the content missing from the third client.” *Id.*, ¶ 70. “In step 616, *the server passes the list of missing content to the third client . . .*, which had been offline.” *Id.*, ¶ 73.

344. A POSA would have found this claim element to be obvious in view of the functionality described above for ’823 Ground 1, claim 1, with respect to Clients A and C for the embodiment in Figure 6, further described in paragraphs 72-74 and 79-80. Sigurdsson, ¶¶ 72-74, 79-80. Namely, Client A could send the “**first file**” to the server, which, in turn, could determine that it is in communication with Client C (after Client C has been offline and resumed communications, for example). Thereafter, the server could send Client C the “list identifying the missing content” referenced in paragraph 73 and, as described in paragraph 79-80, could send missing content (i.e., the “**first file**”) to Client C. *Id.*, ¶¶ 73, 79-80.

345. Transferring the first file to Client C would occur **automatically**, with no user intervention. In addition, the process would be “**responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device,**” because the server would need to check to make sure communications had resumed with Client C and that the server has a copy of the second file before attempting to send it.

346. Moreover, given the fungibility of Sigurdsson’s clients, it would have been obvious to receive a file from Client A at the Server and then automatically transfer the file to Client C in view of the embodiment described in connection with Figure 5, which describes a similar data flow between Clients A and B. Sigurdsson, ¶¶ 52-55 (“In step 508, the server stores the content received from the originating client, and the content is propagated to the second client.”).

347. Sigurdsson’s discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home and work computers and synchronizing both computers to make sure they are updated with the most recent updates on each respective version. For example, Sigurdsson expressly references a client’s work computer and home computer as two exemplary “clients,” and describes example use cases such as a user editing a Word document that is synchronized between the user’s home and work computers:

“For example, a user may edit a [W]ord document file at his home computer. When the user saves the document, the document’s content is automatically extracted and transmitted to the user’s computer at work. When the user searches for the modified document later at his work computer, the content is presented to the user even though the [W]ord document was not previously stored on the user’s work computer.” Sigurdsson, ¶¶ 26-28 (also describing Server 106 as optional “drop box” “conduit” between client devices).

348. A POSA would have found it obvious that in this type of use case, after the copy of the Word document (which may be called “Version 1”) is transmitted from the user’s home computer to the user’s work computer, the user may continue to edit that document on the work computer while the user is at work. Based on Sigurdsson’s bidirectional file synchronization architecture as illustrated in Figure 4, a copy of the updated, edited file at the work computer (call it “Version 2”) would then be transmitted from the work computer, through the Server, and back to the user’s home computer. However, Sigurdsson does not provide details about what would happen at the user’s home computer in that circumstance where “Version 2” of the file is received at the home computer that previously contained “Version 1” of the same file. A POSA would have been motivated to look for solutions to replace old versions of files with new ones as described in this limitation. A POSA would

have found it obvious that one possibility would be for the updated modified file (“Version 2”) to replace the outdated older version of the file (“Version 1”) on the home computer. A POSA would have found it obvious to replace the older version of the file for reasons including the obvious benefit of preventing copies of outdated versions from endlessly accumulating in storage on the user’s computers. Furthermore, Shappell expressly teaches and motivates this feature with its “automatic replication” option, as discussed above in connection with ’823 Ground 1, claim [3c]. In addition, it would have been obvious to combine Sigurdsson with Shappell and Rao for the reasons discussed above in ’823 Ground 1.

2. Claim 3

349. Claim 3 depends from claim 1 and further recites “**wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.**” Claim 3 is not materially different from ’823 element [1i], as shown in the table below.

’823 Claim [1i]	’607 Claim 3
wherein at least one of the server system	The system of claim 1, wherein at least

or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

350. Because claim 3 presents no material difference from '823 element [1i], claim 3 would have been obvious for the same reasons discussed above in connection with the '823.

3. Claim 4

351. Claim 4 depends from claim 1 and further recites “**wherein availability of the version of the first file is presented at the second client device based on the first metadata.**” This limitation would have been obvious for the reasons discussed above in connection with '823 Ground 1, claim [1e].

4. Claim 5

352. Claim 5 depends from claim 1 and further recites “**wherein the server system is caused to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a**

content of the second file stored on the second client device, the copy of the second file being a version of the second file that is generated from the user modifying the content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.” Claim 5 is not materially different from ’823 claim 3, as shown in the table below.

’823 Claim 3	’607 Claim 5
The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	The system of claim 1, wherein the server system is caused to:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being a version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determine that the server system is in communication with the first client device associated with the user; and

automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

353. Because claim 5 presents no material difference from '823 claim 3, claim 5 would have been obvious for the same reasons discussed above in connection with the '823 patent.

5. Claim 7

354. Claim 7 depends from claim 1 and further recites “**wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.**” Claim 7 is not materially different from '823 claim 4, as shown in the table below.

'823 Claim 4	'607 Claim 7
The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

355. Because claim 7 presents no material difference from '823 claim 4, claim 7 would have been obvious for the same reasons discussed above in connection with the '823 patent.

6. Independent Claim 12

356. Claim 12, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 12 (method)
A system comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first	receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
determine that the server system is not in communication with a second client device associated with the user;	determining, by the server system, that the server system is not in communication with a second client device associated with the user;
store the copy of the first file on the server system;	store the copy of the first file on the server system;
automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and	automatically transferring, by the server system, the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.	automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

357. The only substantive difference between claim 1 and claim 12 is that the latter recites “[a] method” for performing the same steps recited in claim 1 that are performed by the server system described in claim 1. Thus, for the reasons discussed above in connection with claim 1, claim 12 would have been obvious.

7. Claim 14

358. Claim 14 depends from claim 12 and further recites “**wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.**” Claim 14 is not materially different from claim 3, as shown in the table below.

Claim 3 (system)	Claim 14 (method)
The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	The method of claim 12, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

359. Because claim 14 presents no material difference from claim 3, claim 14 would have been obvious for the same reasons discussed above in connection with claim 3.

8. Independent Claim 17

360. For reference, claim 17 recites:

[17p] A system comprising:

[17a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[17b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[17c] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration;

[17d] determine that the server system is not in communication with a second client device associated with the user;

[17e] store the copy of the first file on the server system;

[17f] automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

[17g] automatically transfer, over a network, the copy of the first file to the second client device associated with the user, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device,

[17h] wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.

361. Claim 17 is similar to claim 1, as shown in the table below
 (substantively identical limitations appear on the same row):

Claim 1 (system)	Claim 17 (system)
A system comprising:	A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a	receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

first priority greater than a second priority assigned to the copy of the first file;	first priority greater than a second priority assigned to the copy of the first file,
	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration;
determine that the server system is not in communication with a second client device associated with the user;	determine that the server system is not in communication with a second client device associated with the user;
store the copy of the first file on the server system;	store the copy of the first file on the server system;
automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and	automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.	automatically transfer, over a network, the copy of the first file to the second client device associated with the user, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device,

	wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.

- b. [17c]: “receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration;”

362. This claim element is substantively identical to element [1c] in combination with claim 3 and would have been obvious for the reasons discussed above in connection with these claim elements.

- c. **[17h]: “wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.”**

363. Sigurdsson in combination with Shappell discloses “wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.”

364. For the reasons discussed above in ’823 Ground 1, claim [3c], Sigurdsson discloses this claim element. Although the discussion above for ’823 Ground 1, claim [3c], relates to automatically transferring a file from Client C (**second client device**) to Client A (**first client device**), for the reasons discussed above for ’823 Ground 1, claim [3c], it would have been obvious to perform the operations recited in ’823 claim [3c] in the reverse direction to transfer the file from Client A to Client C.

365. For example, in connection with the embodiment depicted in Figure 6, Sigurdsson describes an automated process for (1) detecting when Client C comes back online, (2) passing a list containing metadata to enable Client C to see what

content it is missing, (3) receiving a request from Client C for missing data, and (4) sending the missing data to Client C. The transfer of “Client X Content 410” could be sent automatically to a client that was temporarily offline when it comes back online. Sigurdsson, ¶ 69 (“Step 612 can occur *after the third client comes online, having been offline during the time that the content originated on the first client and was propagated to the second client and to the server*. In step 612, the third client requests a list of missing content from the server. For example, *the Client C 406 requests a list of missing content from the Server 106 when it connects to network that links the server and other clients, such as the Internet.*”). “In step 614, the server determines the content missing from the third client.” *Id.*, ¶ 70. “In step 616, *the server passes the list of missing content to the third client . . .*, which had been offline.” *Id.*, ¶ 73.

366. A POSA would have found this claim element to be obvious in view of the functionality described above for ’823 Ground 1, claim 1, with respect to Clients A and C for the embodiment in Figure 6, further described in paragraphs 72-74 and 79-80. Sigurdsson, ¶¶ 72-74, 79-80. Namely, Client A could send the “**first file**” to the server, which, in turn, could determine that it is in communication with Client C (after Client C has been offline and resumed communications, for example). Thereafter, the server could send Client C the “list identifying the missing content”

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

referenced in paragraph 73 and, as described in paragraph 79-80, could send missing content (i.e., the “**first file**”) to Client C. *Id.*, ¶¶ 73, 79-80.

367. Transferring the first file to Client C would occur **automatically**, with no user intervention. Moreover, given the fungibility of Sigurdsson’s clients, it would have been obvious to receive a file from Client A at the Server and then automatically transfer the file to Client C in view of the embodiment described in connection with Figure 5, which describes a similar data flow between Clients A and B. Sigurdsson, ¶¶ 52-55 (“In step 508, the server stores the content received from the originating client, and the content is propagated to the second client.”).

368. Sigurdsson’s discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home and work computers and synchronizing both computers to make sure they are updated with the most recent updates on each respective version. For example, Sigurdsson expressly references a client’s work computer and home computer as two exemplary “clients,” and describes example use cases such as a user editing a Word document that is synchronized between the user’s home and work computers: “For example, a user may edit a [W]ord document file at his home computer. When the user saves the document, the document’s content is automatically extracted and transmitted to the user’s computer at work. When the user searches for the modified

document later at his work computer, the content is presented to the user even though the [W]ord document was not previously stored on the user's work computer.” Sigurdsson, ¶¶ 26-28 (also describing Server 106 as optional “drop box” “conduit” between client devices).

369. A POSA would have found it obvious that in this type of use case, after the copy of the Word document (which may be called “Version 1”) is transmitted from the user's home computer to the user's work computer, the user may continue to edit that document on the work computer while the user is at work. Based on Sigurdsson's bidirectional file synchronization architecture as illustrated in Figure 4, a copy of the updated, edited file at the work computer (call it “Version 2”) would then be transmitted from the work computer, through the Server, and back to the user's home computer. However, Sigurdsson does not provide details about what would happen at the user's home computer in that circumstance where “Version 2” of the file is received at the home computer that previously contained “Version 1” of the same file. A POSA would have been motivated to look for solutions to replace old versions of files with new ones as described in this limitation. A POSA would have found it obvious that one possibility would be for the updated modified file (“Version 2”) to replace the outdated older version of the file (“Version 1”) on the home computer. A POSA would have found it obvious to replace the older version

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

of the file for reasons including the obvious benefit of preventing copies of outdated versions from endlessly accumulating in storage on the user's computers. Furthermore, Shappell expressly teaches and motivates this feature with its "automatic replication" option, as discussed above in connection with '823 Ground 1, claim [3c]. In addition, it would have been obvious to combine Sigurdsson with Shappell and Rao for the reasons discussed above in '823 Ground 1.

370. Shappell's "automatic replication" option renders obvious implementing Sigurdsson's Server 106 such that, **responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.** Shappell teaches a user-selectable "File Replication" setting where the user can switch between (1) manually downloading files "on an on-demand basis," and (2) automatically downloading files when available, with an "automatic replication" setting. Shappell, ¶¶ 62, 81. When "automatic replication" is enabled, Shappell teaches automatically replacing an "out-of-date file" with an "updated version" on a receiving computer. *Id.*, ¶ 81 ("When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a

preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will *remove the local copy of the out-of-date file. . . . [I]f automatic replication of the shared file is enabled, the local machine obtains an updated version of the file*, as is shown in step 2006 in FIG. 20.”), Figs. 19-20.

371. It would have been obvious to implement Shappell’s “automatic replication” feature as an option for each Sigurdsson client device, including Client C (the “**second client device**”), so that Server 106 would **automatically** transfer to Client C the copy of a file modified at Client A (the “**first client device**”), to ensure that the user at Client C would have the most recently updated file. A POSA would have been motivated to incorporate Shappell’s auto-replication to address Sigurdsson’s concern for each client “to keep its locally stored content synchronized with content from the other clients,” and it easily and obviously would be implemented applying Sigurdsson’s teaching that each client can “maintain an open connection” with the server. Sigurdsson, ¶¶ 84-85.

9. Independent Claim 19

372. Claim 19, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**non-transitory machine-readable media storing instructions**” instead of a “**system.**” As the following table shows, both claims

otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 19 (media)
A system comprising:	One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	determining, by the server system, that the server system is not in communication with a second client device associated with the user;
determine that the server system is not in communication with a second client device associated with the user;	storing, by the server system, the copy of the first file on the server system;
store the copy of the first file on the server system;	automatically transferring, by the server system, the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second
automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second	

client device prior to the copy of the first file being transferred to the second client device; and	to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.	automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

373. The limitation reciting a “**non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system,**” cause operations would have been obvious for the reasons discussed above in connection with claim 13 of the ’823. Thus, for the reasons discussed above for claim 1 and claim 13 of the ’823, this claim would have been obvious.

10. Claim 20

374. Claim 20 depends from claim 19 and further recites “**wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment**

configuration.” Claim 20 is not materially different from claim 3, as shown in the table below.

Claim 3 (system)	Claim 20 (media)
The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	The one or more non-transitory machine-readable media of claim 19, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

375. Because claim 20 presents no material difference from claim 3, claim 20 would have been obvious for the same reasons discussed above in connection with claim 3.

G. '607 Ground 2: Claims 1, 3-5, 7, 12, 14, 17, 19, and 20 of the '607 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell, Rao, and Moromisato.

1. Independent Claim 1

376. Claim 1 of the '607 patent would have been obvious for the reasons discuss above in Ground 1. In addition, it would have been obvious to combine Sigurdsson with Shappell, Rao, and Moromisato and, in view of these references,

claim 1 would have been obvious.

- b. [1g]: “automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.”

377. Moromisato discloses a server-based system in which a change to the server data automatically causes the server to send a command to all clients to modify their data: “Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will***

be the server.” Moromisato, 1:47-54, 14:47-55.

378. Moromisato also teaches the automatic transfer of the modified file copy to another device to replace an older version of the file stored on the other device with the modified copy that includes the content modified by the user. For example, Moromisato teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system ‘sites’. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may either disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

379. It would have been obvious to incorporate this functionality into Sigurdsson’s system in view of the motivation to keep a user’s work and home computers (as well as the computers of other team members, as taught in Moromisato) synchronized automatically as changes are made to avoid the situation in which people view/edit files that are not up-to-date.

380. A POSA would have been motivated to look to references like Moromisato, which also discloses “synchronizer” functionality—specifically, functionality designed to operating in conjunction with the Windows operating system.¹³ Moromisato, 7:9-24 (“The file system RAMP program 304 is part of the collaborative system code and detects changes to files in the OS file system 302. It communicates with the **file synchronizer 308**, as shown schematically by arrow 340, in order to inform the file synchronizer 308 that it needs to synchronize those changes. In particular, ***the Windows XP Pro operating system provides a system event notification system to which clients can subscribe to receive notifications of selected events. These events include changes in the OS file system, such as folder and file edits***, additions and deletions. . . . During the collaborative system start up, the file system RAMP program 304 is invoked so that it can register with the OS event notification system for file notifications indicating ***changes to files*** and file contents in the Windows shell file system 302.”).

381. Like Sigurdsson, Moromisato describes embodiments that use a “server

¹³ Moromisato, assigned to Microsoft, describes functionality related to the then- and now-popular Microsoft Windows operating system. A POSA—and almost everyone else using computers at the time—would have been familiar with the basic file system concepts described in Moromisato, and a POSA would have been motivated to look at Microsoft patents and patent publications at the time as an industry leader in this technical field.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

based” system to implement file synchronization functionality. Moromisato, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. ***Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.***”). A POSA considering Sigurdsson’s similar server-based embodiments would have been motivated to consider the design of Moromisato’s file synchronizer illustrated in Figure 3:

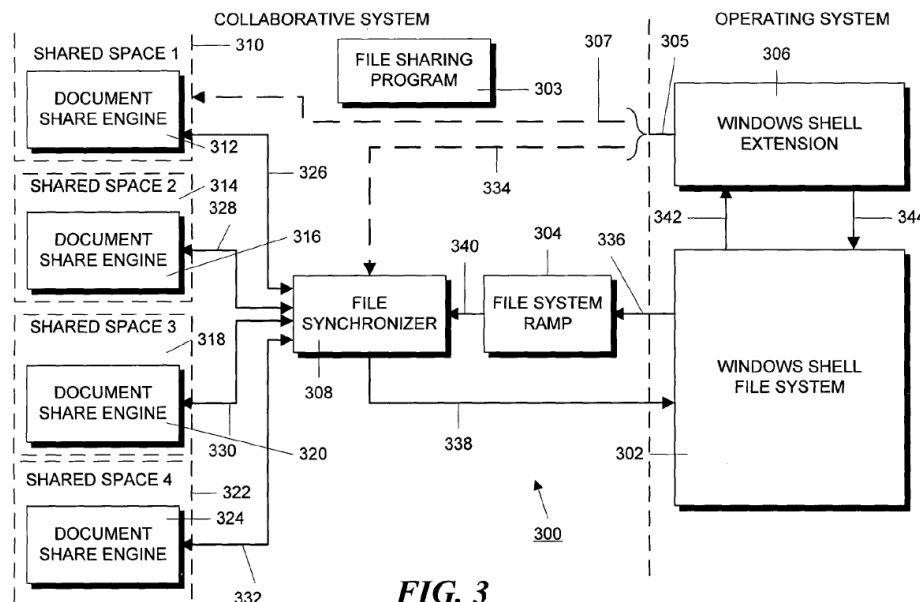


FIG. 3

Id., Fig. 3; see also *id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system and vice

versa.”).

382. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified first electronic file copy to a second client device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44 (“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer **replaces** an older version of the file stored on another device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***.”); see also *id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing).

383. In view of these teachings from Moromisato, it would have been obvious to use the Synchronization Manager in Sigurdsson’s system to automatically transfer the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device,

responsive to (i) resuming communication with the second client device and
(ii) receiving the copy of the first file from the first client device. It would have been
obvious to combine Sigurdsson with Rao, Shappell, and Moromisato for the reasons
above in connection with '823 Ground 2.

2. Claim 4

384. Claim 4 depends from claim 1 and further recites “**wherein availability
of the version of the first file is presented at the second client device based on
the first metadata.**” This claim would have been obvious for the reasons discussed
above in connection with '823 Ground 2, claim [1e].

3. Claim 17

- a. [17h]: “**wherein, responsive to transferring the copy of
the first file to the second client device, an older version
of the first file stored on the second client device is
automatically caused to be replaced with the copy of
the first file such that the copy of the first file is stored
on the second client device in lieu of the older version
of the first file.**”

385. For the reasons discussed above in Ground 1, and the reasons below,
Sigurdsson in combination with Shappell and Moromisato discloses “wherein,
responsive to transferring the copy of the first file to the second client device, an
older version of the first file stored on the second client device is automatically
caused to be replaced with the copy of the first file such that the copy of the first file
is stored on the second client device in lieu of the older version of the first file.”

386. Moromisato discloses a server-based system in which a change to the server data automatically causes the server to send a command to all clients to modify their data: “Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.***” Moromisato, 1:47-54, 14:47-55.

387. Moromisato also teaches the automatic transfer of the modified file copy to another device to replace an older version of the file stored on the other device with the modified copy that includes the content modified by the user. For example, Moromisato teaches: “File synchronizer 308 supports automatic and

manual synchronization of changes between the synchronized file system folder and the collaborative system ‘sites’. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may either disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

388. It would have been obvious to incorporate this functionality into Sigurdsson’s system in view of the motivation to keep a user’s work and home computers (as well as the computers of other team members, as taught in Moromisato) synchronized automatically as changes are made to avoid the situation in which people view/edit files that are not up-to-date.

389. A POSA would have been motivated to look to references like Moromisato, which also discloses “synchronizer” functionality—specifically, functionality designed to operate in conjunction with the Windows operating system.¹⁴ Moromisato, 7:9-24 (“The file system RAMP program 304 is part of the

¹⁴ Moromisato, assigned to Microsoft, describes functionality related to the then- and

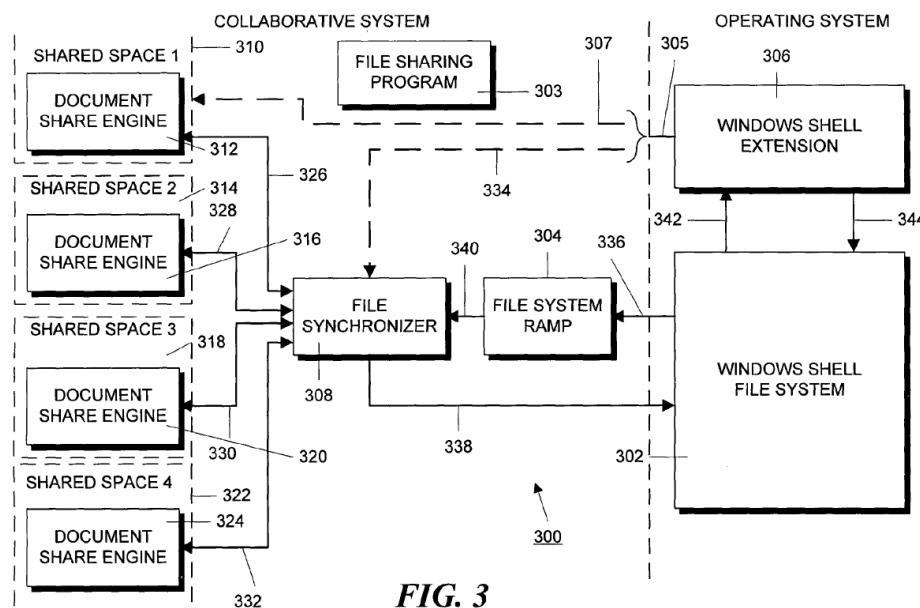
Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

collaborative system code and detects changes to files in the OS file system 302. It communicates with the ***file synchronizer 308***, as shown schematically by arrow 340, in order to inform the file synchronizer 308 that it needs to synchronize those changes. In particular, ***the Windows XP Pro operating system provides a system event notification system to which clients can subscribe to receive notifications of selected events. These events include changes in the OS file system, such as folder and file edits***, additions and deletions. . . . During the collaborative system start up, the file system RAMP program 304 is invoked so that it can register with the OS event notification system for file notifications indicating ***changes to files*** and file contents in the Windows shell file system 302.”).

390. Like Sigurdsson, Moromisato describes embodiments that use a “server based” system to implement file synchronization functionality. Moromisato, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. ***Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.***”). A POSA considering Sigurdsson’s similar

now-popular Microsoft Windows operating system. A POSA—and almost everyone else using computers at the time—would have been familiar with the basic file system concepts described in Moromisato, and a POSA would have been motivated to look at Microsoft patents and patent publications at the time as an industry leader in this technical field.

server-based embodiments would have been motivated to consider the design of
 Moromisato's file synchronizer illustrated in Figure 3:



Id., Fig. 3; *see also id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system and vice versa.”).

391. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified first electronic file copy to a third electronic device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44

(“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer **replaces** an older version of the file stored on another device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***”); see also *id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing).

392. In view of these teachings from Moromisato, it would have been obvious to use the Synchronization Manager in Sigurdsson’s system to automatically transfer the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device.

4. Claims 3, 5, 7, 12, 14, 19, and 20

393. For the reasons discussed above for claim 1, and the reasons discussed above in the corresponding sections of Ground 1, claims 3, 5, 7, 12, 14, 19, and 20 would have been obvious in view of Sigurdsson in combination with Shappell, Rao, and Moromisato.

H. '622 Sigurdsson Grounds: Claims 1, 3-5, 7, 11, 12, 14, 16, and 17 of the '622 Patent Would Have Been Obvious in View of Sigurdsson in Combination with One or More of Shappell, Rao, and Moromisato.

1. Independent Claim 1

394. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[1c] store the copy of the first file on the server system;

[1d] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1e] automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[1f] automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to

receiving the copy of the first file from the first client device.

395. As shown in the table below, claim 1 of the '622 patent is similar to claim 1 of the '607 patent (substantively identical limitations appear on the same row).

'607 Claim 1	'622 Claim 1
1. A system comprising:	1. A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; ¹⁵

¹⁵ The order in which this claim element appears in claim 1 of the '622 patent is altered in the table, but it is otherwise identical to a limitation that appears in claim 1 of the '607 patent. The fact that the limitation appears in a different order than the identical limitation in the '607 patent does not change the required obviousness analysis that I performed above for the '607 patent and, therefore, the analysis

determine that the server system is not in communication with a second client device associated with the user;	
store the copy of the first file on the server system;	store the copy of the first file on the server system;
automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and	automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to . . . receiving the copy of the first file from the first client device.	automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

396. For the reasons discussed above in connection with claim 1 of the '607 patent, claim 1 of the '622 patent would have been obvious.

2. Claim 3

397. This claim is substantively identical to claim 3 of the '607 and, for the reasons discussed above in connection with claim 3 of the '607 patent, would have been obvious.

3. Claim 4

presented in connection with the '607 patent for this limitation applies equally here for this limitation.

398. This claim is substantively identical to claim 4 of the '607 and, for the reasons discussed above in connection with claim 4 of the '607 patent, would have been obvious.

4. Claim 5

399. This claim is substantively identical to claim 5 of the '607 and, for the reasons discussed above in connection with claim 5 of the '607 patent, would have been obvious.

5. Claim 7

400. This claim is substantively identical to claim 7 of the '607 and, for the reasons discussed above in connection with claim 7 of the '607 patent, would have been obvious.

6. Independent Claim 11

401. Claim 11, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**” that performs the steps of the method. Thus, for the reasons discussed above in connection with claim 1, claim 11 would have been obvious.

7. Claim 12

402. Claim 12, which depends from claim 11, would have been obvious for the same reasons discussed above in connection with claim 11 and claim 1 of the '607 patent.

403. '622 element [12a] recites: “initially determining, by the server system, that the server system is not in communication with the second client device associated with the user; and.” This claim limitation would have been obvious for the reasons discussed in connection with '607 element [1d] and the reasons below.

404. Sigurdsson discloses initially determining, by the server system, that the server system is not in communication with the second client device associated with the user.

405. Sigurdsson discloses, in connection with the embodiment of Figure 6, a “sequence diagram showing an illustrative method for sharing and synchronizing content” using the “server architecture of FIG. 4 when a client that is offline comes online.” Sigurdsson, ¶ 61. “In this case, ***the system 400 automatically detects that a particular client is offline.*** In one implementation, ***determination of whether a client is online or offline*** can be made by monitoring polling by the client. If no request is received from a client, then the client is assumed to be offline. Similarly, the Server 106 can maintain a table of client IDs and whether each client is online or offline. When a client comes online, it can notify the Server 106 that it is online, and the Server 106 can update its table of clients with the online status.” *Id.* Thus, Sigurdsson discloses an initial determination by the Server that a client is offline followed by a determination (when notified by the client) that communication with

the client has resumed.

406. For example, in connection with the embodiment in Figure 6, Sigurdsson discloses determining whether “Client C” is offline (which would, of course, mean that Client C would not be in communication with the Server). Sigurdsson, ¶ 66. As explained above, Sigurdsson’s various “clients” are fungible, so the same functionality could be applied with respect to Clients A, B, or C in Sigurdsson. See, e.g., *id.*, ¶ 51 (“The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from **any the** [sic] **Clients 102, 104 or 406. The drop-box, or storage, capabilities of the Server 106 can be used, for example, if a particular client is offline when the first attempt is made to deliver new content to it.** Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the Client C 406 can have communicate [sic] with the Server 106 in a manner similar to the method used by the Clients A or B.”).

407. ’622 element [12b] recites: “automatically transferring, by the server system, the copy of the first file to the second client device to replace the older version of the first file stored on the second client device with the copy of the first file, responsive to (i) resuming communication with the second client device, (ii) determining that the server system is in communication with the second client

device, (iii) and receiving the copy of the first file from the first client device.” This claim limitation would have been obvious for the reasons discussed in connection with ’607 element [1g] and the reasons below.

408. As discussed above in connection with element [12a], Sigurdsson discloses an initial determination by the Server that a client is offline followed by a determination (when notified by the client) that communication with the client has resumed. Therefore, Sigurdsson discloses “determining that the server system is in communication with the second client device” after “resuming communication with the second client device.” The remaining claim limitations would have been obvious for the reasons discussed above in connection with ’607 element [1g].

8. Claim 14

409. This claim is substantively identical to claim 14 of the ’607 and, for the reasons discussed above in connection with claim 14 of the ’607 patent, would have been obvious.

9. Independent Claim 16

410. Claim 16, for purposes of my analysis, is substantially similar to claim 1 except that it recites one or more “**non-transitory machine-readable media storing instructions**” instead of a “**system**.” For the reasons discussed above for claims 1 and 13 of the ’823, this claim would have been obvious.

10. Claim 17

411. Claim 17 depends from claim 16 and further recites “**wherein at least one of the first computer system or the second computer system comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the first computer system or the second computer system assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.**” Claim 17 is not materially different from claim 3 and, because claim 17 presents no material difference from claim 3, claim 17 would have been obvious for the same reasons discussed above in connection with claim 3.

I. ’942 Ground 1: Claims 1, 3, 4, 10, 12 and 13 of the ’942 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell.

1. Independent Claim 1

412. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a first electronic device, associated with a user, configured to:

[1b] receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first

electronic file;

[1c] determine whether the first electronic device is in communication with a third electronic device;

[1d] automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;

[1e] receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;

[1f] determine whether the first electronic device is in communication with the second electronic device; and

[1g] automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,

[1h] wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and

[1i] wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second

electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

b. [1p]: “A system comprising:”

413. The preamble of claim 1 recites “[a] **system.**” I understand that the preamble of a claim generally does not limit the claim, but to the extent that the preamble provides a claim limitation, it is disclosed by Sigurdsson, for the reasons discussed above in connection with ’823 element [1p].

c. [1a]: “a first electronic device, associated with a user, configured to:”

414. Sigurdsson discloses a first electronic device associated with a user that is configured.

415. For example, “Server 106” in Figure 4 (and as described in connection with Figure 9 and elsewhere in Sigurdsson) is a “**first electronic device**” that may be “**configured.**” For example, Sigurdsson discloses that “System 900” depicted in Figure 9 “may be included in . . . Server 106.” Sigurdsson, ¶ 104. System 900 includes “a **Processor 910**, a **Memory 920**, a **Storage Device 930**, and Input/Output Devices 940” and “[t]he **Processor 910 is capable of processing instructions for execution within the System 900.**” *Id.*, ¶ 105. “The Processor 910 is capable of **processing instructions stored in the Memory 920 or on the Storage Device 930** to

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

display graphical information for a user interface on the Input/Output Devices 940.”

Id. Sigurdsson further discloses, in connection with System 900:

The apparatus can be implemented in a **computer program** product tangibly embodied in an information carrier, e.g., in a machine-readable storage device or in a propagated signal, **for execution by a programmable processor**; and **method steps can be performed by a programmable processor executing a program of instructions to perform functions of the described implementations** by operating on input data and generating output. The described features can be implemented advantageously in one or more computer programs that are executable on a programmable system including at least one programmable processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. **A computer program is a set of instructions that can be used, directly or indirectly, in a computer to perform a certain activity or bring about a certain result.**

Id., ¶ 109; see also *id.*, ¶ 110 (“Generally, **a processor will receive instructions** and data from a read-only memory or a random access memory or both. The essential elements of a computer are **a processor for executing instructions** and one or more memories for storing instructions and data.”). Thus, Server 106 is “**configured**” to operate according to the instructions stored in Memory 920 and/or Storage Device 930 and executed by the one or more processors discussed above.

416. Sigurdsson discloses that Server 106 may be “**associated with a user.**” For example, Server 106 may be associated with the user because the Server can store and authenticate a user ID for each user. Sigurdsson, ¶ 49 (“The Authentication Manager 334 can provide security features that prevent unauthorized clients from

using the Server 106 for storing and requesting content. Authentication can be based on, for example, a system for verifying user IDs and passwords. The Authentication Manager 334 can check the credentials of a client when the client connects to the Server 106. If a client supplies invalid credentials, the Authentication Manager can transmit an error message to the client.”); *see also id.*, ¶ 27 (“The **Server 106** can provide a temporary storage location, or ‘drop box’, for *a user’s* files, documents, web pages, or the contents of these documents.”).

- d. [1b]: “receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;”

417. Sigurdsson discloses receiving, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file.

418. To start, Sigurdsson discloses a **first electronic device** (“Server 106”) that “receive[s], via a first application at the first electronic device, a copy of a **modified first electronic file from a second application at a second electronic**

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

device.” “Server 106” in Figure 4 includes a “Synchronization Manager 332,” which a POSA would understand to be a “**first application.**” Sigurdsson, ¶ 48. Sigurdsson further discloses that Synchronization Manager 332 is an application comprising one or more computer program instructions that, when executed by one or more processors in Server 106, would cause Server 106 to perform operations. *Id.* As shown in Sigurdsson Figure 3 and described in the text, such as in paragraph 48, the server contains only two programs/applications (the Synchronization Manager 332 and Authentication Manager 334). The Authentication Manager 334 would not be understood by a POSA to perform the relevant claimed functions, since Authentication Manager 334 is described only as handling authentication/security functions (per paragraph 49). Given that Authentication Manager 334 would not be understood to handle the relevant claimed functions, it would have been obvious to a POSA that Synchronization Manager 332 is used to perform the relevant functions discussed herein; and there would have been obvious advantages and a natural design choice in having that single application responsible for the relevant functions, for simple and efficient design of the relevant software. Sigurdsson explains that Server 106 “can be implemented in a computer system that includes a back-end component, such as a data server, or that includes a middleware component, such as an application server or an Internet

server[.]” *Id.*, ¶ 112. Synchronization Manager 332 includes “a Space Quota 342 and a Throughput Threshold 344” that are used to control the storage of data to the server. *Id.*, ¶ 48. Sigurdsson explains that Server 106 “can be implemented in a computer system that includes a back-end component, such as a data server, or that includes a middleware component, such as an **application** server or an Internet server.” *Id.*, ¶ 112. A POSA would understand that an application server could be used to service requests, such as receiving and storing files from clients to the server, using a “**first application**” (Synchronization Manager 332).¹⁶

419. Client A (“**second electronic device**”) and Client C (“**third electronic device**”) are in communication with Server 106 (“**first electronic device**”), as shown in Figure 4. See Sigurdsson, ¶ 51 (“Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the *Client C 406 can have communicate* [sic] *with the Server 106 in a manner similar to the method used by the Clients A or B.*”).

420. Client A includes a “**second application**” executable on Client A in the

¹⁶ It would have been obvious to a POSA to combine one or more functionalities into a single application or to alternatively keep them in separate applications. For example, it would have been obvious to include the functionality for sending and receiving data between clients and the server and the functionality for managing storage on the server in a single application on the server, as this would have been well within the knowledge of a POSA.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

form of an HTTPS client/server: “The **Client A 102** can include both data and **applications**. . . . The **applications** include an Event Listener 322, a File to Text/HTML Converter 324, and **a hypertext transfer protocol secure (HTTPS) Client/server 326**.” Sigurdsson, ¶ 47. A POSA would understand that the HTTPS Client/server application is the application that transfers content from Client A to the server via, for example, an HTTP “post” command. *E.g., id.*, ¶ 54 (“For example, referring to FIG. 4, the HTTPS Client/server 326 of the Client A 102 can transmit an HTTP post command containing the Client X Content 410 to the Server 106.”).

421. Client A may “**automatically**” transfer a “**first electronic file**” using the “**second application**” (HTTPS Client/server) that posts content from the client to the server. For example, the sequence in Figure 6 is described as “showing an illustrative method for sharing and synchronizing content using [the] . . . server architecture of FIG. 4[.]” Sigurdsson, ¶ 61. An exemplary “**first electronic file**” disclosed by Sigurdsson as residing on Client A is a “very large” “maintenance manual.” *Id.*, ¶ 58. When a user edits a file (e.g., the maintenance manual), the file may be automatically transferred from the second application to the server if another client that needs to receive the updated file is offline. *Id.*, ¶ 62 (“The system 400 can enable . . . synchronization using . . . ‘drop box’ techniques. . . . [I]n order for the system 400 to supply the content to an offline client, **the server can be used as a**

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

drop box for the content until the offline client comes online again.”).

422. Paragraphs 63 and 68 in Sigurdsson explain that content may be sent directly from Client A to Server 106 when a user “edit[s]” a document (e.g., a maintenance manual). Sigurdsson, ¶¶ 63 (“***Processing can start*** in step 602 ***when an event occurs***. This step 602 may be similar to the step 202 of FIG. 2, where ***an event is generated in response to an action performed by a user, such as, saving a file or document***, accessing a web page, or opening a file or application. ***For example, referring to FIG. 4, a user at the Client A 102 may edit and save a document on his work computer.***”), 68 (Client A “***sends the data directly to the server***, as shown by the dashed line pointing from the step 606 to the step 610”). It would have been obvious to send the maintenance manual to Server 106 “***when content is changed*** on the Client A 102.” *Id.* ¶ 84 (“***The Client A 102 posts Client A Content 108 to the Server 106 when content is changed on the Client A 102.*** Posting of the Client A Content 108 occurs, for example, whenever the user saves a file or document ***[P]osting can occur as soon as an event is generated by an action***, such as saving a file. For example, a client can maintain an open connection with a server and post ***as soon as an event is generated.***”).

423. As one way to implement this feature, Sigurdsson describes that an event listener listens for events such as, e.g., when a user modifies a file, and, in

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

response to such events, the Client/server application will send the modified files to the Server. Sigurdsson, ¶¶ 32-33 (“For example, . . . a user at the Client A 102 may edit and save his resume on his home computer. This action may generate an event that is detected by an event listener. . . . For example, after the event triggered by an action is detected, the content from the file associated with the action can be extracted. The extracted content can be the text or images included in the file that was saved, accessed or *modified*.”); *id.*, ¶ 45 (“The Client A 102 posts the Client A Content 108 to the Server 106 *after content is changed* on the Client A 102.”); *id.*, ¶ 47 (“The *Event Listener 322 listens for and detects user actions*, such as saving a file, deleting a file, or viewing a web page, *that may require the associated . . . Client A Content 108 to be posted to the Server 106*.”). Thus, Server 106 (“**first electronic device**”) may “**receive, via a first application**” (Synchronization Manager 332) “**at the first electronic device**” (Server 106), “**a copy of a modified first electronic file**” (maintenance manual) “**from a second application**” (HTTPS Client/server) “**at a second electronic device associated with the user**” (Client A). It would have been obvious to apply these teachings of automatically sending content from Client A to Server 106 when the content is changed to the exemplary “**first electronic file**” (maintenance manual) when it is modified, “**wherein the modified first electronic file copy is automatically received from the second**

application responsive to the user modifying a content of the first electronic file” (the modified maintenance manual is automatically received by Server 106 from the HTTPS Client/server 326 at Client A responsive to the user modifying the maintenance manual).

424. The content transferred to the Server may include a full copy of the file modified by the user. Sigurdsson, ¶ 65 (“In step 606, the content originating on the first client is posted [to the Server]. *The content can include a copy of the document that the user saves . . .*”); see also *id.*, ¶ 8 (“[E]xtracting the content of the first file may include *generating a copy of the first file that retains the first file’s original file formatting.*”); *id.*, ¶ 28 (“The Client A Content 108, which can include *a copy of the user’s file*, is sent automatically to the Server 106.”).

425. As a result of Client A posting the content to the server, “[t]he Client A Content 108 is received by and stored in the Server 106.” Sigurdsson, ¶ 85. For example, Sigurdsson discloses that its server may store a modified first electronic copy to a memory device associated with the server. *Id.*, ¶ 48 (“The **Server 106 can contain a Storage 330**, a Synchronization Manager 332, and an Authentication Manager 334. The **Storage 330** can contain the Content A, and Content B through Content N 336 that has been received from the user’s various clients.”); *id.*, ¶ 74 (“the content may be stored at the server”). Sigurdsson further discloses: “The

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Server 106 serves as a **temporary storage location** that can facilitate synchronization between the user's clients." *Id.*, ¶ 83; see also *id.*, ¶ 85 ("The Client A Content 108 is received by and **stored in the Server 106.**"). Sigurdsson further discloses "System 900" in Figure 9 and explains that "**the System 900 may be included in** either or all of the Client A 102, the Client B 104, and **the Server 106.**" *Id.*, ¶ 104. In connection with the embodiment in Figure 9, Sigurdsson discloses the use of a memory device: "Storage devices suitable for tangibly embodying computer program instructions and data include **all forms of non-volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.**" *Id.*, ¶ 110.

426. Sigurdsson further teaches that Server 106 **receives a copy of a modified first electronic file** from a client device, such as Client A. Sigurdsson, ¶¶ 28 ("The Client A Content 108, **which can include a copy of the user's file**, is sent automatically to the Server 106."), 54 ("In step 506, the content originating on the client is posted to the server. **The content can include a copy of the document (or the content of the document) that was saved by the user[.]**"), 65 (discussing copy of file saved by user), 85 (same), 103; see also *id.*, ¶ 8 ("[E]xtracting the content of the first file may include generating a copy of the first file that retains the

first file's original file formatting.”). I note that although some embodiments refer to peer-to-peer sending of files (e.g., from Client A to Client B), Sigurdsson describes that files may be sent directly to the server system by a client, including in the embodiment described in Figures 4-7. For example, Client A may “**send[] the data directly to the server**, as shown by the dashed line pointing from the step 606 to the step 610. For example, referring to FIG. 4, the **Client X Content 410 can be sent directly to the Server 106 . . .**” *Id.*, ¶ 68; *see also id.*, ¶¶ 84-85 (describing that Client A posts content to the Server).

427. A POSA would have appreciated and recognized Sigurdsson's teachings of the obvious advantages of a client/server arrangement. The server can stay online while user computers periodically go offline, so that Server **106** beneficially provides a “drop-box” for “a user's files” to “**facilitate sharing this information between the user's computers by storing the shared content if the target computer is offline.**” Sigurdsson, ¶¶ 27-29, 61 (server tracks clients' online/offline status). A server may also have more available bandwidth relative to clients for transferring data. *Id.*, ¶ 79. Sigurdsson makes clear the peer-to-peer and client/server pathways between the user's multiple client devices are interchangeable, rendering it obvious to synchronize through the server. *Id.*, ¶ 51. For example, Sigurdsson explains that each client may include an “HTTPS

Client/server” that may interchangeably be used for synchronization with Server 106 (*id.*, ¶ 54 (“HTTPS Client/server 326 of the Client A 102 can transmit an HTTP post command containing the Client X Content 410 to the Server 106.”)) **or** synchronization with other clients (*id.*, ¶47 (“The HTTPS Client/server can also be used as a server to provide content to peer clients, which is described in more detail below.”)). Even when clients can communicate peer-to-peer, Sigurdsson teaches that they can still also use Server 106 for content sharing. *Id.*, ¶¶51, 79 (“Direct content sharing among peer clients can make use of available resource capacities at the clients, thus reducing the bandwidth traffic at the Server 106. ***The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from any the [sic] Clients 102, 104 or 406.***”).

428. Lastly, Sigurdsson discloses that its client devices and Server 106 may be “**associated with a user.**” Sigurdsson, ¶ 31 (“The Server 106 can facilitate synchronization of content among the user’s computers.”). Client A is described as, for example, a “user’s home computer” and Client B is described as, for example, a “user’s work computer.” *Id.*, ¶¶ 28-29, 55. Figure 4 depicts Client C as a user’s mobile device. *Id.*, Fig. 4, ¶ 27 (“A user may access many computers including, for example, computers at home, computers at work, and mobile computers, such as personal digital assistants (PDAs) and cell phones.”). Clients A, B, and C are

associated with the same user. *Id.*, ¶ 58 (“For example, a user can store a very large document (e.g., a maintenance manual) on the Client A 102, and at a short time later, the same user empties a digital camera on the Client B 104. The same day, the user may go to work and update a small office memo on the Client C 406.”); *see also id.*, ¶ 2 (“[U]sers want to have access to their digital information whether they are on their home computer, using a laptop at work, or on the road with a wireless personal digital assistant.”). The Server is also associated with the user because the Server can store and authenticate a user ID for each user. *Id.*, ¶ 49 (“The Authentication Manager 334 can provide security features that prevent unauthorized clients from using the Server 106 for storing and requesting content. Authentication can be based on, for example, a system for verifying user IDs and passwords. The Authentication Manager 334 can check the credentials of a client when the client connects to the Server 106. If a client supplies invalid credentials, the Authentication Manager can transmit an error message to the client.”); *id.*, ¶ 27 (“The **Server 106** can provide a temporary storage location, or “drop box”, for **a user’s** files, documents, web pages, or the contents of these documents.”).

e. [1c]: “determine whether the first electronic device is in communication with a third electronic device;”

429. Sigurdsson discloses or renders obvious determining whether the first electronic device is in communication with a third electronic device.

430. Sigurdsson discloses, in connection with the embodiment of Figure 6, a “sequence diagram showing an illustrative method for sharing and synchronizing content” using the “server architecture of FIG. 4 when a client that is offline comes online.” Sigurdsson, ¶ 61. “In this case, ***the system 400 automatically detects that a particular client is offline.*** In one implementation, ***determination of whether a client is online or offline can be made by monitoring polling by the client.*** If no request is received from a client, then the client is assumed to be offline. Similarly, the Server 106 can maintain a table of client IDs and whether each client is online or offline. ***When a client comes online, it can notify the Server 106 that it is online,*** and the Server 106 can update its table of clients with the online status.” *Id.* The notification sent to the server from a client that comes online allows the server to determine that the server is in communication with the first client device associated with the user. As a specific example, in connection with the embodiment described in Figure 6, Sigurdsson discloses that Client C may be offline. *Id.*, ¶¶ 66, 73. In view of the above, it would have been obvious to use Server 106 to “**determine whether the first electronic device**” (Server 106) “**is in communication with a third electronic device**” (Client C).

431. Sigurdsson’s various “clients” are fungible, so it would have been obvious to apply the functionality in this limitation to Clients A, B, or C. See

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Sigurdsson, ¶ 51 (“The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from ***any the*** [sic] ***Clients 102, 104 or 406. The drop-box, or storage, capabilities of the Server 106 can be used, for example, if a particular client is offline when the first attempt is made to deliver new content to it.*** Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the Client C 406 can have communicat[ion] with the Server 106 in a manner similar to the method used by the Clients A or B.”). Sigurdsson explains that “Clients A and B 102, 104 can contain similar components and can accept and transmit content in a similar manner. For example, both clients can include Files 346, HTML Files 348, an Index 350, an Event Listener 352, a File to Text/HTML Converter 354 and an HTTPS Client/server 356.” *Id.*, ¶ 50. Because Clients A, B, and C are described as operating similarly, they all could post/push data/files to the Server in the same way using the same components. *See also id.*, Fig. 3 (depicting both Client A and Client B as having identical event listeners and HTTPS client/servers). In addition, the functionality related to transferring data from Client A to the Server and then from the Server to other clients is described generically as applying to any combination of the clients and the server. *See id.*, ¶ 68 (“For example, referring to FIG. 4, the Client X Content 410 can be sent directly to the Server 106 The Server 106 can then transmit the content to the remaining

client peers, such as the Clients B and C.”). Thus, the functionality for synchronizing files related to any particular client described in Sigurdsson could apply to any other client as well (the clients are fungible), and the server may detect whether Clients A, B, or C are online or offline and in communication or out of communication with the server.

- f. [1d]: “**automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;**”

432. Sigurdsson in combination with Shappell discloses or renders obvious automatically sending, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device.

433. Sigurdsson discloses or renders obvious “**send[ing], via the first application**” (Synchronization Manager 332), “**the modified first electronic file copy**” (copy of the modified maintenance manual) “**to a third application**” (HTTPS Client/server) “**at the third electronic device**” (Client C). As discussed above for

claim [1b], Client C may communicate with Server 106 using the same HTTPS Client/server used by Clients A and B. Sigurdsson, ¶ 51 (“Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the ***Client C 406 can have communicate*** [sic] ***with the Server 106 in a manner similar to the method used by the Clients A or B.***”). Sigurdsson explains in connection with the embodiment in Figure 6 that Server 106 may send the modified maintenance manual received from Client A to Client C. *Id.*, ¶ 80 (“In step 626, the server sends the missing content to the third client. For example, referring to FIG. 4, the ***Server 106 sends the Client X Content 410 to the Client C 406.***”).

434. Sigurdsson also discloses or renders obvious that the first electronic file copy is sent “**responsive to the determination that the first electronic device is in communication with the third electronic device.**” As discussed above in connection with claim [1c], Sigurdsson discloses or renders obvious determining whether the server is in communication with Client C. Only after this determination is made, and only in response to this determination, is the file sent to Client C. Sigurdsson, ¶ 69 (“Step 612 can occur ***after the third client comes online***, having been offline during the time that the content originated on the first client and was propagated to the second client and to the server. . . . For example, the ***Client C 406 requests a list of missing content from the Server 106 when it connects to network***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

that links the server and other clients, such as the Internet.”). Thus, it would have been obvious for the server to first determine that it is in communication with Client C before transmitting content to Client C, using these methods.

435. It would have been obvious in view of Sigurdsson in combination with Shappell to “**automatically**” send the file “**responsive to receiving the modified first electronic file copy from the second electronic device.**” Sigurdsson’s Server in Figure 4 includes a “Synchronization Manager” that “synchroniz[e] content stored on and shared between” different clients (e.g., “Client A” and “Client C”). Sigurdsson, ¶ 48 (“The Synchronization Manager 332 can use the Time Stamps 338 to synchronize content stored on and shared between the Client A 102 and the Client B 104.”). A POSA would understand that “synchronization” in this context means “version comparisons of copies of . . . files to ensure they contain the same data.” EX1021, Microsoft Computer Dictionary (5th ed. 2002), 506 (entry for “synchronization”); *see also id.* (“In handheld computer, the process of updating or backing up the data on a handheld computer to the linked software applications on a desktop computer. ***Data changes made on the desktop computer may also be copied to the handheld during synchronization.***”). Thus, a POSA would have understood and found it obvious that Sigurdsson’s reference to “synchroniz[ing]” content stored on and shared between clients would have encompassed using the

Synchronization Manager to automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device.

436. Sigurdsson explains that Client A content received at the Server could be automatically transferred by the Server to Client B or any other client. Sigurdsson, ¶ 68 (“For example, referring to FIG. 4, the Client X Content 410 can be sent directly to the Server 106 ***The Server 106 can then transmit the content to the remaining client peers, such as the Clients B and C.***”). As an example, in connection with the embodiment depicted in Figure 6, Sigurdsson describes an automated process for (1) detecting when Client C comes back online, (2) passing a list containing metadata to enable Client C to see what content it is missing, (3) receiving a request from Client C for missing data, and (4) sending the missing data to Client C. The transfer of “Client X Content 410” could be sent automatically when a temporarily offline client comes back online. *Id.*, ¶ 69. “In step 614, the server determines the content missing from the third client.” *Id.*, ¶ 70. “In step 616, ***the server passes the list of missing content to the third client . . . ,*** which had been offline.” *Id.*, ¶ 73.

437. Sigurdsson also discloses sending the **entire** file once it has been modified. Sigurdsson, ¶ 8 (“Additionally, extracting the content of the first file may include **generating a copy of the first file that retains the first file’s original file formatting.**”); *id.*, ¶ 28 (“The Client A Content 108, which can include **a copy of the user’s file**, is sent automatically to the Server 106.”); *id.*, ¶ 54 (“In step 506, the content originating on the client is posted to the server. The content can include **a copy of the document** (or the content of the document) that was saved by the user as well as metadata that describes information, such as the type of data, its source, when it was saved, and when it was posted to the server.”); *id.*, ¶ 65 (“The content can include **a copy of the document** that the user saves as well as metadata that describes the type of data, the client that generated the data, and when the data was last changed.”).

438. Sigurdsson’s discussion of synchronization would have motivated a POSA to consider solutions to the problem of a user wanting to keep an up-to-date version of a file on their home and work computers and synchronizing both computers to make sure they are updated as soon as possible with the most recent version. For example, Sigurdsson expressly discloses automatically transferring a modified file from one client to the server (as discussed above in connection with claim [1b]), and other embodiments describe automatically transferring the modified

file from the server to another client when the latter client connects to the server. Sigurdsson, ¶ 31 (“When the Client A 102 (e.g., the user’s home computer) connects to the Server 106, Client B Content 112 originating from the Client B 104 is automatically copied to the Client A 102, as indicated by arrow 116. . . . Similarly, whenever the Client B 104 (e.g., the user’s work computer) connects to the Server 106, any missing Client A Content 108 is copied automatically to the Client B 104, as indicated by arrow 118.”). Sigurdsson further discloses in another embodiment that “a client can maintain an open connection with a server and post as soon as an event is generated.” *Id.*, ¶ 84. It would have been obvious in view of these disclosures to automatically transfer a modified file received from Client A to Server 106, and then automatically transfer the modified file, via an “open connection,” to Client C as soon as the file is received at the server.

439. In addition, or in the alternative, it would have been obvious to practice this limitation in view of Sigurdsson in combination with Shappell. Shappell expressly discloses automatically transferring a modified file to another client after receiving the modified file from an originating client. For example, Shappell discloses an automatic replication feature that automatically copies files added to a group’s “shared space” to the user’s computer “when they are added to the shared space.” Shappell, ¶ 81 (“When a modification of a file contained in a shared space

occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file.*** This will change the visual representation of the file to a ‘missing’ file within the shared space folder when the application setting does not automatically update files, as is shown in step 2010 in FIG. 20. On the other hand, ***if automatic replication of the shared file is enabled, the local machine obtains an updated version of the file,*** as is shown in step 2006 in FIG. 20.”). It would have been obvious to apply this concept from Shappell to Sigurdsson’s systems and methods to automatically send a modified file from the server to Client C when it has arrived at the server.

440. Sigurdsson discloses concerns about the limited memory on Server 106 and the need to manage the amount of data stored on Server 106. For example, Synchronization Manager 332 includes “a Space Quota 342 and a Throughput Threshold 344” that are used to control the storage of data to the server. Sigurdsson, ¶ 48. A POSA would have been motivated in view of these concerns to automatically transfer an incoming, updated file from Server 106 to any other client device that may need the file as quickly as possible to avoid unnecessary use of Server 106’s memory if transfer of the file is delayed. It would have been obvious

to combine Sigurdsson with Shappell for the reasons discussed above in connection with '823 Ground 1, claim [1c].

441. Transferring the second file to Client A would occur **automatically**, with no user intervention. In addition, the process would be “**responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device**,” because the server would need to check to make sure that Client C is online and that it actually has a copy of the file before attempting to send it.

442. Moreover, given the fungibility of Sigurdsson's clients, it would have been obvious to receive a file from Client A at the Server and then automatically transfer the file to Client C in view of the embodiment described in connection with Figure 5, which describes a similar data flow between Clients A and B. Sigurdsson, ¶¶ 52-55 (“In step 508, the server stores the content received from the originating client, and the content is propagated to the second client.”).

443. Sigurdsson's discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home and work computers and synchronizing both computers to make sure they are updated with the most recent updates on each respective version. For example,

Sigurdsson expressly references a client's work computer and home computer as two exemplary "clients," and describes example use cases such as a user editing a Word document that is synchronized between the user's home and work computers: "For example, a user may edit a [W]ord document file at his home computer. When the user saves the document, the document's content is automatically extracted and transmitted to the user's computer at work. When the user searches for the modified document later at his work computer, the content is presented to the user even though the [W]ord document was not previously stored on the user's work computer." Sigurdsson, ¶¶ 26-28 (also describing Server 106 as optional "drop box" "conduit" between client devices).

444. A POSA would have found it obvious that in this type of use case, after the copy of the Word document (which may be called "Version 1") is transmitted from the user's home computer to the user's work computer, the user may continue to edit that document on the work computer while the user is at work. Based on Sigurdsson's bidirectional file synchronization architecture as illustrated in Figure 4, a copy of the updated, edited file at the work computer (call it "Version 2") would then be transmitted from the work computer, through the Server, and back to the user's home computer. However, Sigurdsson does not provide details about what would happen at the user's home computer in that circumstance where "Version 2"

of the file is received at the home computer that previously contained “Version 1” of the same file. A POSA would have been motivated to look for solutions to replace old versions of files with new ones as described in this limitation. A POSA would have found it obvious that one possibility would be for the updated modified file (“Version 2”) to replace the outdated older version of the file (“Version 1”) on the home computer. A POSA would have found it obvious to replace the older version of the file for reasons including the obvious benefit of preventing copies of outdated versions from endlessly accumulating in storage on the user’s computers. Furthermore, Shappell expressly teaches and motivates this feature with its “automatic replication” option, as discussed further below.

445. Shappell’s “automatic replication” option further renders obvious implementing Sigurdsson’s Server 106 to **automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device.** Shappell teaches a user-selectable “File Replication” setting where the user can switch between (1) manually downloading files “on an on-demand basis,” and (2) automatically downloading files when available, with an “automatic replication”

setting. Shappell, ¶¶ 62, 81. When “automatic replication” is enabled, Shappell teaches automatically replacing an “out-of-date file” with an “updated version” on a receiving computer. *Id.*, ¶ 81 (“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file***. This will change the visual representation of the file to a ‘missing’ file within the shared space folder when the application setting does not automatically update files, as is shown in step 2010 in FIG. 20. On the other hand, ***if automatic replication of the shared file is enabled, the local machine obtains an updated version of the file***, as is shown in step 2006 in FIG. 20.”), Figs. 19-20.

446. It would have been obvious to implement Shappell’s “automatic replication” feature as an option for each Sigurdsson client device, including Client C (the “**third electronic device**”), so that Server 106 would **automatically** transfer to Client C the copy of a file modified at Client A (the “**second electronic device**”), to ensure that the user at Client C would have the most recently updated file. A POSA would have been motivated to incorporate Shappell’s auto-replication to address Sigurdsson’s concern for each client “to keep its locally stored content

synchronized with content from the other clients,” and it easily and obviously would be implemented applying Sigurdsson’s teaching that each client can “maintain an open connection” with the server. Sigurdsson, ¶¶ 84-85.

- g. [1e]: “receive, via the first application, a copy of a **modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;**”

447. Sigurdsson discloses receiving, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file.

448. For the reasons discussed above in connection with claim [1c], Sigurdsson’s various “clients” are fungible, so it would have been obvious to implement the functionality performed by Client A in claim [1b] with respect to the copy of the modified maintenance manual (“**copy of a modified first electronic file**”) but instead using the HTTPS Client/server (“**third application**”) at Client C (“**third electronic device**”) to transfer a “**copy of a modified second electronic file.**” For example, an exemplary “**copy of a modified second electronic file**” at

Client C is described as, e.g., “a small office memo” in Sigurdsson. Sigurdsson, ¶
58 (“[T]he user may go to work and update a small office memo on the Client C
406.”).

**h. [1f]: “determine whether the first electronic device is
in communication with the second electronic device;
and”**

449. Sigurdsson discloses determining whether the first electronic device is
in communication with the second electronic device.

450. For the reasons discussed above in connection with claim [1c],
Sigurdsson’s various “clients” are fungible, so it would have been obvious to
implement the functionality described in claim [1c] for determining whether the first
electronic device is in communication with the third electronic device but instead
for “**determin[ing] whether the first electronic device is in communication with
the second electronic device.**” For example, Sigurdsson discloses determining
whether “Client B” is offline instead of Client C (and, of course, it would have been
obvious to also detect whether the server is in communication with Client A
 (“**second electronic device**”), given that Clients A and B include similar
components and perform similar functions). Sigurdsson, ¶ 50 (“Clients A and B
102, 104 can contain similar components and can accept and transmit content in a
similar manner. For example, both clients can include Files 346, HTML Files 348,

an Index 350, an Event Listener 352, a File to Text/HTML Converter 354 and an HTTPS Client/server 356.”). Thus, for the same reasons above for claim [1c], Sigurdsson discloses that its server may detect whether Clients A, B, or C are online or offline and in communication or out of communication with the server.

- i. **[1g]: “automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,”**

451. Sigurdsson discloses automatically sending, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device.

452. For the reasons discussed above in connection with claim [1c], Sigurdsson’s various “clients” are fungible. Thus, for the reasons discussed above in connection with claims [1c] and [1d], it would have been obvious to automatically transfer the modified “**second electronic file copy**” (small office memo) from the server, via the server’s “**first application**,” to the “**second application**” (HTTPS Client/server at Client A) “**responsive to the determination that the first**

electronic device” (Server 106) “is in communication with the second electronic device” (Client A) “and responsive to receiving the modified second electronic file copy from the third electronic device” (Client C).

- j. [1h]: **“wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and”**

453. Sigurdsson in combination with Shappell discloses, in response to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file.

454. For the reasons discussed above in claim [1d], Sigurdsson in combination with Shappell discloses and renders obvious **“sending the modified first electronic file copy to the third electronic device.”**

455. Sigurdsson discloses that a **“first electronic file copy”** may be **“stored on the third electronic device.”** Sigurdsson, ¶ 81 (“In step 628, the content received by the third client is stored on the client’s local storage device. For

example, referring to FIG. 4, ***the Client X Content 410 received by the Client C 406 is stored on the client's local storage device.***”). Sigurdsson discloses “System 900” in Figure 9 and explains that “***the System 900 may be included in*** either or all of the ***Client A 102, the Client B 104,*** and the Server 106.” *Id.*, ¶ 104. It would have been obvious that similar storage could have been included in Client C, given that Clients A, B, and C are described in Sigurdsson as being fungible for the reasons discussed above in connection with claim [1c]. In addition, it would have been obvious that, instead of being a mobile device as shown in Figure 4 that may have potentially had limited memory in November 2008, Client C could have also been a computer with memory large enough to store a large file, such as an updated maintenance manual, for the reasons discussed above in connection with claim [1c] (regarding the various client devices being fungible). *Id.*, ¶ 45 (“The Client A 102 and the Client B 104 each represent a computer used by a user including, for example, a personal computer at home, a computers at work, and portable computers, such as a laptop computer, a PDA, and a cell phone.”). As noted above, Sigurdsson explains that its clients are fungible in that they contain the same components and may implement the same functionality with respect to sending and receiving files from the server. *See, e.g., id.*, ¶¶ 50-51. Thus, it would have been obvious to a POSA that the functionality described with respect to Client A could be implemented in Client C

and vice versa. *Id.*

456. It would have been obvious in view of Sigurdsson in combination with Shappell that, **“responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device”** could **“automatically”** be **“caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file.”**

457. Sigurdsson’s Server in Figure 4 includes a “Synchronization Manager” that “synchronize[s] content stored on and shared between” different clients (e.g., “Client A” and “Client C”). See Sigurdsson, ¶ 48. For the reasons discussed above in claim [1d], a POSA would understand that “synchronization” in this context means “version comparisons of copies of . . . files to ensure they contain the same data.” EX1021, Microsoft Computer Dictionary (5th ed. 2002), 506 (entry for “synchronization”); see also *id.* (“In handheld computer, the process of updating or backing up the data on a handheld computer to the linked software applications on a desktop computer. ***Data changes made on the desktop computer may also be copied to the handheld during synchronization.***”). Thus, a POSA would have understood and found it obvious that Sigurdsson’s reference to “synchroniz[ing]”

content stored on and shared between clients would have encompassed using the Synchronization Manager to automatically transfer an updated copy of a file from one client device through Sigurdsson's server to another client device associated with the user to replace an older version of the file stored on the latter client device.

458. Transferring the **modified first electronic file** to Client C (**third electronic device**) would occur **automatically**, with no user intervention. Moreover, given the fungibility of Sigurdsson's clients, it would have been obvious to receive a file from Client A at the Server and then automatically transfer the file to Client C in view of the embodiment described in connection with Figure 5, which describes a similar data flow between Clients A and B. Sigurdsson, ¶¶ 52-55 ("In step 508, the server stores the content received from the originating client, and the content is propagated to the second client.").

459. Sigurdsson's discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home and work computers and synchronizing both computers to make sure they are updated with the most recent updates on each respective version. For example, Sigurdsson expressly references a client's work computer and home computer as two exemplary "clients," and describes example use cases such as a user editing a Word document that is synchronized between the user's home and work computers:

“For example, a user may edit a [W]ord document file at his home computer. When the user saves the document, the document’s content is automatically extracted and transmitted to the user’s computer at work. When the user searches for the modified document later at his work computer, the content is presented to the user even though the [W]ord document was not previously stored on the user’s work computer.” Sigurdsson, ¶¶ 26-28 (also describing Server 106 as optional “drop box” “conduit” between client devices).

460. A POSA would have found it obvious that in this type of use case, after the copy of the Word document (which may be called “Version 1”) is transmitted from the user’s home computer to the user’s work computer, the user may continue to edit that document on the work computer while the user is at work. Based on Sigurdsson’s bidirectional file synchronization architecture as illustrated in Figure 4, a copy of the updated, edited file at the work computer (call it “Version 2”) would then be transmitted from the work computer, through the Server, and back to the user’s home computer. However, Sigurdsson does not provide details about what would happen at the user’s home computer in that circumstance where “Version 2” of the file is received at the home computer that previously contained “Version 1” of the same file. A POSA would have been motivated to look for solutions to replace old versions of files with new ones as described in this limitation. A POSA would

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

have found it obvious that one possibility would be for the updated modified file (“Version 2”) to replace the outdated older version of the file (“Version 1”) on the home computer. A POSA would have found it obvious to replace the older version of the file for reasons including the obvious benefit of preventing copies of outdated versions from endlessly accumulating in storage on the user’s computers. Furthermore, **Shappell** expressly teaches and motivates this feature with its “automatic replication” option, as discussed further below.

461. Shappell’s “automatic replication” option further renders obvious implementing Sigurdsson’s Server 106 such that, **responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file.** Shappell teaches a user-selectable “File Replication” setting where the user can switch between (1) manually downloading files “on an on-demand basis,” and (2) automatically downloading files when available, with an “automatic replication” setting. Shappell, ¶¶ 62, 81. When “automatic replication” is enabled, Shappell teaches automatically replacing an “out-of-date file” with an “updated version” on a receiving computer. *Id.*, ¶ 81

(“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file. . . . [I]f automatic replication of the shared file is enabled, the local machine obtains an updated version of the file***, as is shown in step 2006 in FIG. 20.”), Figs. 19-20.

462. It would have been obvious to implement Shappell’s “automatic replication” feature as an option for each Sigurdsson client device, including Client C (the “**third electronic device**”), so that Server 106 would **automatically** transfer to Client C the copy of a file modified at Client A, to ensure that the user at Client C would have the most recently updated file. A POSA would have been motivated to incorporate Shappell’s auto-replication to address Sigurdsson’s concern for each client “to keep its locally stored content synchronized with content from the other clients,” and it easily and obviously would be implemented applying Sigurdsson’s teaching that each client can “maintain an open connection” with the server. Sigurdsson, ¶¶ 84-85.

- k. [1i]: “wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.”

463. Sigurdsson discloses wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

464. As noted above for claim [1c], Sigurdsson explains that its clients are fungible. Thus, for the reasons discussed above in connection with element [1h], it would have been obvious to implement the functionality described in element [1h] but with respect to the “**second electronic file copy**” instead of the first electronic file copy and the “**second electronic device**” instead of the third electronic device.

2. Claim 3

465. Claim 3 depends from claim 1 and further recites “**wherein the first electronic device is configured to store, via the first application, the modified**

first electronic file copy to a memory device associated with the first electronic device.” The limitations of claim 3 are disclosed by and obvious over the disclosures in Sigurdsson.

466. Sigurdsson discloses that its Server 106 is “**configured to store, via the first application**” (Synchronization Manager 332) “**the modified first electronic file copy**” (modified copy of the maintenance manual) “**to a memory device associated with the first electronic device**” (Storage 330). Sigurdsson, ¶ 48 (“The *Server 106 can contain a Storage 330*, a Synchronization Manager 332, and an Authentication Manager 334. The *Storage 330* can contain the Content A, and Content B through Content N 336 that has been received from the user’s various clients.”); *id.*, ¶ 74 (“the content may be stored at the server”). Sigurdsson further discloses: “The Server 106 serves as a *temporary storage location* that can facilitate synchronization between the user’s clients.” *Id.*, ¶ 83; *see also id.*, ¶ 85 (“The Client A Content 108 is received by and *stored in the Server 106.*”). Sigurdsson also expressly discloses “System 900” in Figure 9 and explains that “*the System 900 may be included in* either or all of the Client A 102, the Client B 104, and *the Server 106.*” *Id.*, ¶ 104. In connection with the embodiment in Figure 9, Sigurdsson discloses the use of a memory device: “Storage devices suitable for tangibly embodying computer program instructions and data include *all forms of non-*

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.” Id., ¶ 110.

467. As discussed above, the “**first application**” includes Server 106’s “Synchronization Manager 332,” which includes “a Space Quota 342 and a Throughput Threshold 344” that are used to control the storage of data to the server. Sigurdsson, ¶ 48. Thus, a POSA would have found it to be obvious that the Synchronization Manager 332 (“**first application**”) manages not only the sending and receiving of files to and from clients but also the storage of such files to Server 106, and that storage to Server 106 would therefore be “**via the first application**” (Synchronization Manager 332). Also, as discussed above, this functionality could be included as part of an “application server” that could be used to service requests to store data to the server. *Id.*, ¶ 112. A POSA would understand that an application server could be used to service requests, such as storing data from clients to the Server.

3. Claim 4

468. Claim 4 depends from claim 3 and further recites “**wherein the modified first electronic file copy is stored on the memory device associated**

with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.” The limitations of claim 4 are disclosed by and obvious over the disclosures in Sigurdsson.

469. As discussed above in connection with claim 3, Sigurdsson discloses that **“the modified first electronic file copy is stored on the memory device associated with the first electronic device.”**

470. Sigurdsson’s discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home computer, work computer, and Server 106 and the issue of synchronizing all three devices to make sure they are updated with the most recent updates on each respective version. For example, Sigurdsson expressly references a client’s work computer and home computer as two exemplary “clients,” and describes example use cases such as a user editing a Word document that is synchronized between the user’s home and work computers: “For example, a user may edit a [W]ord document file at his home computer. When the user saves the document, the document’s content is automatically extracted and transmitted to the user’s computer at work. When the user searches for the modified document later at his work computer, the content is presented to the user even though the [W]ord document was not previously

stored on the user's work computer." Sigurdsson, ¶¶ 26-28 (also describing Server 106 as optional "drop box" "conduit" between client devices).

471. A POSA would have found it obvious that in this type of use case, after the copy of the Word document (which may be called "Version 1") is transmitted from the user's home computer to the user's work computer via Server 106, the user may continue to edit that document on the work computer while the user is at work. Based on Sigurdsson's bidirectional file synchronization architecture as illustrated in Figure 4, a copy of the updated, edited file at the work computer (call it "Version 2") would then be transmitted from the work computer, through Server 106, and back to the user's home computer. However, Sigurdsson does not provide details about what would happen at the user's home computer or Server 106 in that circumstance where "Version 2" of the file is received at Server 106 and the home computer that previously contained "Version 1" of the same file. A POSA would have been motivated to look for solutions to replace old versions of files with new ones as described in this limitation. A POSA would have found it obvious that one possibility would be for the updated modified file ("Version 2") to replace the outdated older version of the file ("Version 1") on Server 106 and the home computer. A POSA would have found it obvious to replace the older version of the file for reasons including the obvious benefit of preventing copies of outdated

versions from endlessly accumulating in storage on the user's computers and Server 106.

472. Sigurdsson further discloses that the "Synchronization Manager 332" in Server 106 "contains a Space Quota 342" that may be used to "purge" certain client content "whenever the storage space quota for the client is reached." Sigurdsson, ¶ 48. Sigurdsson further discloses that content "may no longer exist on the server if it was deleted, for example, due to **retention rules**, storage quotas, or an unforeseen loss of data." *Id.*, ¶ 96. Perhaps the most obvious example of such a retention rule would be to replace older versions of files with newer versions, in view of Sigurdsson's dual concerns of maintaining file currency across multiple client devices "when content is changed" (i.e., as soon as possible) (*id.*, ¶ 84) and not wasting disk space on the server (*id.*, ¶ 96).

473. As another example, it would have been obvious to use the storage quota functionality of Server 106 to delete an older version of a file and replace it with an updated version of the file. For example, Sigurdsson discloses:

Storage quotas on a server can limit the amount of data stored by a user. For example, the Server 106 may have a configurable storage quota for each client it serves. ***The Client A 102 (e.g., a user's work computer) may transmit content to the Server 106 until the client exceeds its storage quota.*** In some implementations, the Server 106 may have a FIFO system for handling a client's transmitted content relative to its quota. ***If the client's quota is exceeded, the new content can be accepted, but the oldest content can be deleted.*** If the content is deleted

before another client (e.g., a user's home computer that has been offline for months) is able to receive it, then the client can be forced to obtain the content elsewhere. In this example, the client (e.g., user's home computer) can obtain the content directly from his work computer. Alternatively, the Server may request the content from the client that generated the content and transmit the content to the requesting client.

Sigurdsson, ¶ 98. In this example, one obvious use case could be for Server 106 to receive and synchronize multiple versions of a single file (**modified first electronic file copy**) from Client A until Client A's storage quota is reached, requiring deletion of an old version of the file. In this use case, **the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.**

474. Furthermore, Shappell expressly teaches and motivates this feature with its “automatic replication” option, as discussed further below.

475. Shappell's “automatic replication” option further renders obvious implementing Sigurdsson's Server 106 **wherein the modified first electronic file copy is stored on the memory device associated with Server 106 (first electronic device) in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.** Shappell teaches a user-selectable “File Replication” setting where the user can switch between (1) manually downloading files “on an on-demand basis,” and (2)

automatically downloading files when available, with an “automatic replication” setting. Shappell, ¶¶ 62, 81. When “automatic replication” is enabled, Shappell teaches automatically replacing an “out-of-date file” with an “updated version” on a receiving computer. *Id.*, ¶ 81 (“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file. . . . [I]f automatic replication of the shared file is enabled, the local machine obtains an updated version of the file***, as is shown in step 2006 in FIG. 20.”), Figs. 19-20.

476. It would have been obvious that, when implementing Shappell’s “automatic replication” feature, Server 106 should delete all old versions of a file stored on Server 106 after receiving an updated version of the file, given that no user device would ever want or need an older version of the file (they are deleted on the client devices) and in view of Sigurdsson’s storage quota concerns for Server 106. A POSA would have been motivated to incorporate Shappell’s auto-replication to address Sigurdsson’s concern for each client “to keep its locally stored content synchronized with content from the other clients,” and it easily and obviously would

be implemented applying Sigurdsson's teaching that each client can "maintain an open connection" with the server. Sigurdsson, ¶¶ 84-85.

477. A POSA would have further been motivated to replace an older version of a file on Server 106 with a newer version in view of Moromisato, which discloses a server-based system in which a change to the server data automatically causes the server to send a command to all clients to modify their data. As part of this process, Moromisato discloses that an older version of a file is replaced on the server: "Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the***

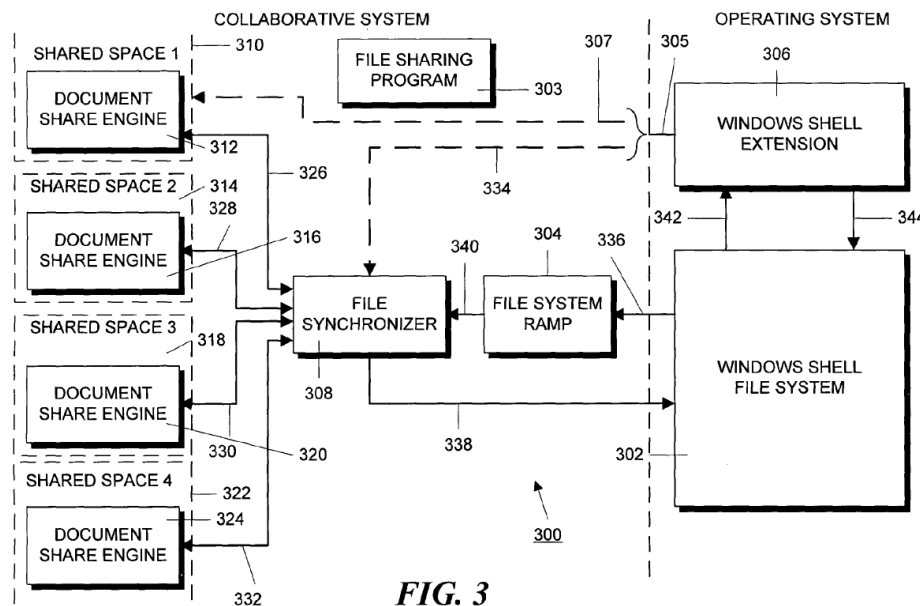
Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

case of server-based systems, that source will be the server.” Moromisato, 1:47-54, 14:47-55.

478. Moromisato further teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system ‘sites’. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may either disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

479. Like Sigurdsson, Moromisato describes embodiments that use a “server based” system to implement file synchronization functionality. *Id.*, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. ***Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.***”). A POSA considering Sigurdsson’s similar server-

based embodiments would have been motivated to consider the design of
 Moromisato's file synchronizer illustrated in Figure 3:



Moromisato, Fig. 3; *see also id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system 302 and vice versa.”).

480. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified file copy to a user device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44 (“Each shared space is an

instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer **replaces** an older version of the file stored on the device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***”); see also *id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing). It would have been obvious in view of these disclosures to replace an older version of a file on Server 106 with a newer version received from a client.

4. Independent Claim 10

481. Claim 10, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 10 (system)
A system comprising:	A method comprising:
a first electronic device, associated with a user, configured to:	

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;	receiving, via a first application at a first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with a user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;
determine whether the first electronic device is in communication with a third electronic device;	determining whether the first electronic device is in communication with a third electronic device;
automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;	automatically sending, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;
receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;	receiving, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;
determine whether the first electronic device is in communication with the second electronic device; and	determining whether the first electronic device is in communication with the second electronic device; and

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,	automatically sending, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,
wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and	wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and
wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.	wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

482. Because claim 10 presents no material difference from claim 1, claim 10 would have been obvious for the same reasons discussed above in connection

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

with claim 1.

5. Claim 12

483. This claim is substantively identical to claim 3 and, for the reasons discussed above in connection with claim 3, would have been obvious.

6. Claim 13

484. This claim is substantively identical to claim 4 and, for the reasons discussed above in connection with claim 4, would have been obvious.

J. '942 Ground 2: Claims 1, 3, 4, 10, 12 and 13 of the '942 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Moromisato.

1. Independent Claim 1

485. Claim 1 of the '942 patent would have been obvious for the reasons discuss above in Ground 1. In addition, it would have been obvious to combine Sigurdsson with Shappell and Moromisato and, in view of these references, claim 1 would have been obvious.

- b. [1d]: “automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and**

**responsive to receiving the modified first electronic file
copy from the second electronic device”**

486. Moromisato discloses a server-based system in which a change to the server data automatically causes the server to send a command to all clients to modify their data: “Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.***” Moromisato, 1:47-54, 14:47-55.

487. Moromisato also teaches the automatic transfer of the modified file copy to another device to replace an older version of the file stored on the other device with the modified copy that includes the content modified by the user. For

example, Moromisato teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system ‘sites’. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may*** either ***disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

488. It would have been obvious to incorporate this functionality into Sigurdsson’s system in view of the motivation to keep a user’s work and home computers (as well as the computers of other team members, as taught in Moromisato) synchronized automatically as changes are made to avoid the situation in which people view/edit files that are not up-to-date.

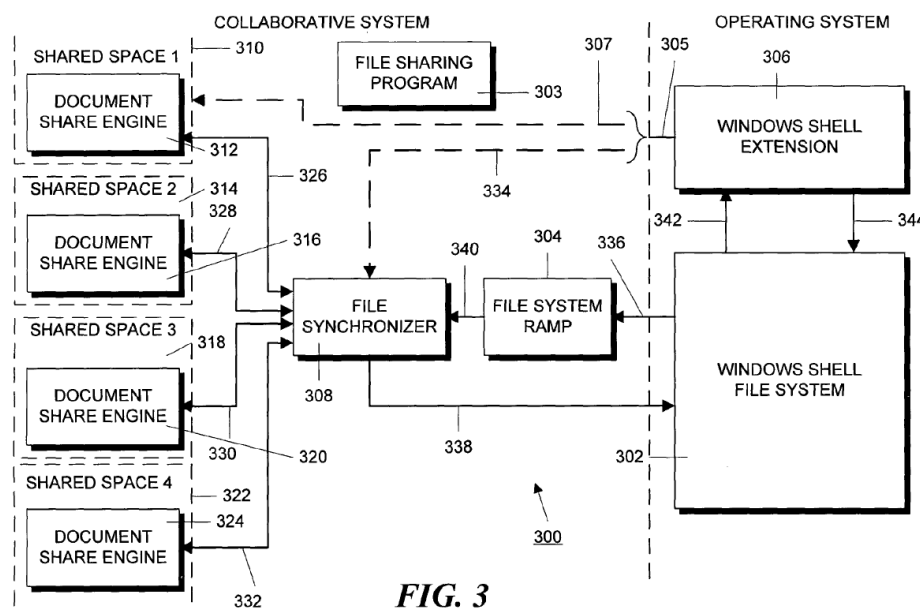
489. A POSA would have been motivated to look to references like Moromisato, which also discloses “synchronizer” functionality—specifically, functionality designed to operating in conjunction with the Windows operating

system.¹⁷ Moromisato, 7:9-24 (“The file system RAMP program 304 is part of the collaborative system code and detects changes to files in the OS file system 302. It communicates with the **file synchronizer 308**, as shown schematically by arrow 340, in order to inform the file synchronizer 308 that it needs to synchronize those changes. In particular, **the Windows XP Pro operating system provides a system event notification system to which clients can subscribe to receive notifications of selected events. These events include changes in the OS file system, such as folder and file edits**, additions and deletions. . . . During the collaborative system start up, the file system RAMP program 304 is invoked so that it can register with the OS event notification system for file notifications indicating **changes to files** and file contents in the Windows shell file system 302.”).

490. Like Sigurdsson, Moromisato describes embodiments that use a “server based” system to implement file synchronization functionality. Moromisato, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer.

¹⁷ Moromisato, assigned to Microsoft, describes functionality related to the then- and now-popular Microsoft Windows operating system. A POSA—and almost everyone else using computers at the time—would have been familiar with the basic file system concepts described in Moromisato, and a POSA would have been motivated to look at Microsoft patents and patent publications at the time as an industry leader in this technical field.

Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.”). A POSA considering Sigurdsson’s similar server-based embodiments would have been motivated to consider the design of Moromisato’s file synchronizer illustrated in Figure 3:



Id., Fig. 3; *see also id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system and vice versa.”).

491. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified file copy

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

to a user device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44 (“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer **replaces** an older version of the file stored on the device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***”); *see also id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing).

492. In view of these teachings from Moromisato, it would have been obvious to use the Synchronization Manager in Sigurdsson’s system to automatically transfer a copy of a file to a client device associated with the user to replace an older version of the file stored on the client device, responsive to (i) determining that the server system is in communication with the client device and (ii) receiving a copy of the file from another client device. It would have been obvious to combine Sigurdsson with Moromisato and Shappell for the reasons discussed above in

connection with '823 Ground 2.

- c. **[1g]: “automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device”**

493. Because, as discussed in Ground 1, Sigurdsson’s clients are fungible and implement the same functionality, and for the reasons above in connection with element [1d], this claim limitation would have been obvious.

2. Claim 4

494. Claim 4 depends from claim 3 and further recites “**wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.**” For the reasons discussed above in Ground 1, and the reasons below, the limitations of claim 4 are disclosed by and would have been obvious in view of the disclosures in Sigurdsson in combination with Shappell and Moromisato.

495. A POSA would have further been motivated to replace an older version of a file on Server 106 with a newer version in view of Moromisato, which discloses a server-based system in which a change to the server data automatically causes the

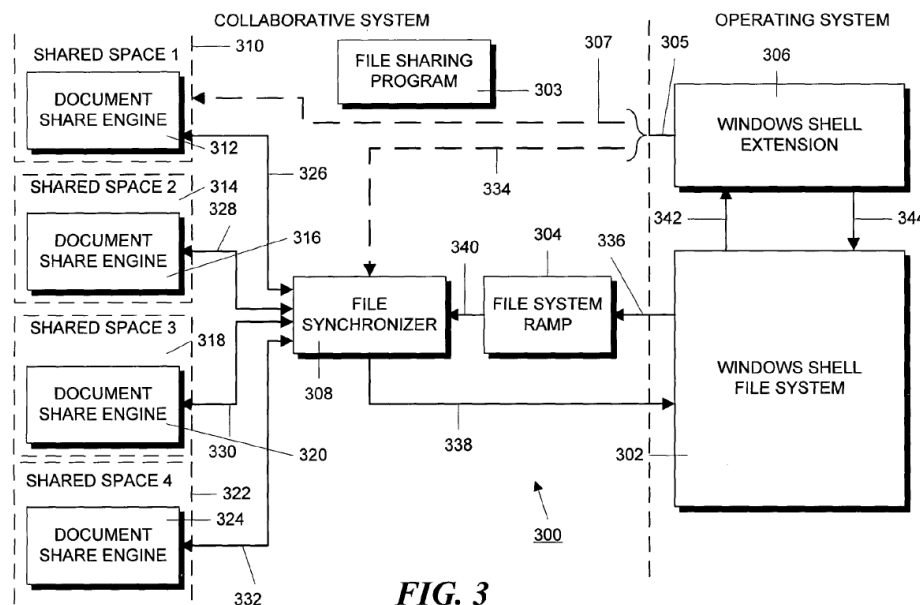
Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

server to send a command to all clients to modify their data. As part of this process, Moromisato discloses that an older version of a file is replaced on the server: “Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.***” Moromisato, 1:47-54, 14:47-55.

496. Moromisato further teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system ‘sites’. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file

synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may either disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

497. Like Sigurdsson, Moromisato describes embodiments that use a “server based” system to implement file synchronization functionality. Moromisato, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. ***Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.***”). A POSA considering Sigurdsson’s similar server-based embodiments would have been motivated to consider the design of Moromisato’s file synchronizer illustrated in Figure 3:



Id., Fig. 3; *see also id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system and vice versa.”).

498. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified file copy to a user device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44 (“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

replaces an older version of the file stored on the device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***”); see also *id.*, 21:19 (listing “[*file edit (replace)*” as a file-change command that the file synchronizer would be tasked with processing). It would have been obvious in view of these disclosures to replace an older version of a file on Server 106 with a newer version received from a client.

3. Claims 3, 10, 12 and 13

499. For the reasons discussed above for claim 1, and the reasons discussed above in the corresponding sections of Ground 1, claims 3, 10, 12 and 13 would have been obvious in view of Sigurdsson in combination with Shappell and Moromisato.

K. ’561 Ground 1: Claims 1, 3, 4, 8, 10 and 11 of the ’561 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell.

1. Independent Claim 1

500. For reference, claim 1 recites:

[1p] A system, comprising:

[1a] a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:

[1b] receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and

[1c] wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and

[1d] wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and

[1e] automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;

[1f] wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.

b. [1p]: “A system comprising:”

501. The preamble of claim 1 recites “[a] system.” I understand that the preamble of a claim generally does not limit the claim, but to the extent that the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

preamble provides a claim limitation, it is disclosed by Sigurdsson, for the reasons
discussed above in connection with '823 claim [1p].

- c. [1a]: “a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:”

502. Sigurdsson discloses a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured.

503. For example, “Server 106” in Figure 4 (and as described in connection with Figure 9 and elsewhere in Sigurdsson) is a “**first electronic device**” that may be “**configured.**” For example, Sigurdsson discloses that “System 900” depicted in Figure 9 “may be included in . . . Server 106.” Sigurdsson, ¶ 104. System 900 includes “a **Processor 910**, a **Memory 920**, a **Storage Device 930**, and Input/Output Devices 940” and “[t]he **Processor 910 is capable of processing instructions for execution within the System 900.**” *Id.* ¶ 105. “The Processor 910 is capable of **processing instructions stored in the Memory 920 or on the Storage Device 930** to display graphical information for a user interface on the Input/Output Devices 940.” *Id.* Sigurdsson further discloses, in connection with System 900:

The apparatus can be implemented in a **computer program** product tangibly embodied in an information carrier, e.g., in a machine-readable storage device or in a propagated signal, **for execution by a programmable processor**; and **method steps can be performed by a programmable processor executing a program of instructions to perform functions of the described implementations** by operating on input data and generating output. The described features can be implemented advantageously in one or more computer programs that are executable on a programmable system including at least one programmable processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. **A computer program is a set of instructions that can be used, directly or indirectly, in a computer to perform a certain activity or bring about a certain result.**

Id. ¶ 109; see also *id.* ¶ 110 (“Generally, **a processor will receive instructions** and data from a read-only memory or a random access memory or both. The essential elements of a computer are **a processor for executing instructions** and one or more memories for storing instructions and data.”). Thus, Server 106 is “**configured**” to operate according to the instructions stored in Memory 920 and/or Storage Device 930 and executed by the one or more processors discussed above.

504. Sigurdsson discloses that Server 106 may be “**associated with a user.**” For example, Server 106 may be associated with the user because the Server can store and authenticate a user ID for each user. Sigurdsson, ¶ 49 (“The Authentication Manager 334 can provide security features that prevent unauthorized clients from using the Server 106 for storing and requesting content. Authentication can be based on, for example, a system for verifying user IDs and passwords. The Authentication

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Manager 334 can check the credentials of a client when the client connects to the Server 106. If a client supplies invalid credentials, the Authentication Manager can transmit an error message to the client.”); *see also id.* ¶ 27 (“The **Server 106** can provide a temporary storage location, or ‘drop box’, for *a user’s* files, documents, web pages, or the contents of these documents.”).

505. Client A (“**second electronic device**”) and Client C (“**third electronic device**”) are in communication with Server 106 (“**first electronic device**”), as shown in Figure 4. *See id.* ¶ 51 (“Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the **Client C 406 can have communicate** [sic] **with the Server 106 in a manner similar to the method used by the Clients A or B.**”). Each of Sigurdsson’s client devices are described as being “**associated with a user.**” *Id.*, ¶ 31 (“The Server 106 can facilitate synchronization of content among the user’s computers.”). Client A is described as, for example, a “user’s home computer” and Client B is described as, for example, a “user’s work computer.” *Id.* ¶¶ 28-29, 55. Figure 4 depicts Client C as a user’s mobile device. *Id.*, Fig. 4, ¶ 27 (“A user may access many computers including, for example, computers at home, computers at work, and mobile computers, such as personal digital assistants (PDAs) and cell phones.”). Clients A, B, and C are associated with the same user. *Id.* ¶ 58 (“For example, a user can store a very large document (e.g.,

a maintenance manual) on the Client A 102, and at a short time later, the same user empties a digital camera on the Client B 104. The same day, the user may go to work and update a small office memo on the Client C 406.”); *see also id.* ¶ 2 (“[U]sers want to have access to their digital information whether they are on their home computer, using a laptop at work, or on the road with a wireless personal digital assistant.”).

506. A POSA would have appreciated and recognized Sigurdsson’s teachings of the obvious advantages of a client/server arrangement. The server can stay online while user computers periodically go offline, so that Server 106 beneficially provides a “drop-box” for “a user’s files” to “***facilitate sharing this information between the user’s computers by storing the shared content if the target computer is offline.***” Sigurdsson, ¶¶ 27-29, 61 (server tracks clients’ online/offline status). A server may also have more available bandwidth relative to clients for transferring data. *Id.* ¶ 79. Sigurdsson makes clear the peer-to-peer and client/server pathways between the user’s multiple client devices are interchangeable, rendering it obvious to synchronize through the server. *Id.* ¶ 51. For example, Sigurdsson explains that each client may include an “HTTPS Client/server” that may interchangeably be used for synchronization with Server 106 (*id.* ¶ 54 (“HTTPS Client/server 326 of the Client A 102 can transmit an HTTP post

command containing the Client X Content 410 to the Server 106.”)) **or** synchronization with other clients (*id.* ¶ 47 (“The HTTPS Client/server can also be used as a server to provide content to peer clients, which is described in more detail below.”)). Even when clients can communicate peer-to-peer, Sigurdsson teaches that they can still also use Server 106 for content sharing. *Id.*, ¶¶ 51, 79 (“Direct content sharing among peer clients can make use of available resource capacities at the clients, thus reducing the bandwidth traffic at the Server 106. ***The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from any the [sic] Clients 102, 104 or 406.***”).

507. “Server 106” in Figure 4 includes a “Synchronization Manager 332,” which a POSA would understand to be a “**first application.**”¹⁸ Sigurdsson, ¶ 48. Sigurdsson further discloses that Synchronization Manager 332 is an application comprising one or more computer program instructions that, when executed by one or more processors in Server 106, would cause Server 106 to perform operations. *Id.*, ¶ 48. As shown in Sigurdsson Figure 3 and described in the text,

¹⁸ It would have been obvious to a POSA to combine one or more functionalities into a single application or to alternatively keep them in separate applications. For example, it would have been obvious to include the functionality for sending and receiving data between clients and the server and the functionality for managing storage on the server in a single application on the server, as this would have been well within the knowledge of a POSA.

such as in paragraph 48, the server contains only two programs/applications (the Synchronization Manager 332 and Authentication Manager 334). The Authentication Manager 334 would not be understood by a POSA to perform the relevant claimed functions, since Authentication Manager 334 is described only as handling authentication/security functions (per paragraph 49). Given that Authentication Manager 334 would not be understood to handle the relevant claimed functions, it would have been obvious to a POSA that Synchronization Manager 332 is used to perform the relevant functions discussed herein; and there would have been obvious advantages and a natural design choice in having that single application responsible for the relevant functions, for simple and efficient design of the relevant software. Sigurdsson explains that Server 106 “can be implemented in a computer system that includes a back-end component, such as a data server, or that includes a middleware component, such as an application server or an Internet server” *Id.* ¶ 112. Synchronization Manager 332 includes “a Space Quota 342 and a Throughput Threshold 344” that are used to control the storage of data to the server. *Id.* ¶ 48. Sigurdsson explains that Server 106 “can be implemented in a computer system that includes a back-end component, such as a data server, or that includes a middleware component, such as an **application** server or an Internet server” *Id.* ¶ 112. A POSA would understand that an application server could

be used to service requests, such as receiving and storing files from clients to the server, using a “**first application**” (Synchronization Manager 332).POSA

508. **Selective execution** of the **first application** may be implemented based on metadata values associated with content to determine whether the content should be transmitted from the server to a client device as part of synchronization. Sigurdsson, ¶ 9 (“The system also includes a data table to store the content extracted from the file on the first client and the metadata value, an interface to receive from a second client a request for content that is associated with one or more metadata values within a specified metadata value range, and *a selection module to initiate transmission of the content to the second client if the metadata value associated with the content is within the specified metadata value range.*”). At a minimum, selective execution would have been obvious in view of these disclosures in combination with the knowledge of a POSA. For example, a POSA would understand that any computer processor selectively executes application instructions by switching back and forth between application code and operating system code. As another example, an application is selectively executed only when necessary and when called upon to execute certain functionality, such as the functionality recited in this claim limitation.

- d. [1b]: “receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and”

509. Sigurdsson discloses receiving from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file.

510. Sigurdsson discloses a **first electronic device** (“Server 106”) that “receive[s] from a second application executable on the second electronic device a copy of a modified first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file.”

Client A includes a “**second application**” executable on Client A in the form of an HTTPS client/server: “The *Client A 102* can include both data and **applications**. . . .

The **applications** include an Event Listener 322, a File to Text/HTML Converter 324, and **a hypertext transfer protocol secure (HTTPS) Client/server 326**.”

Sigurdsson, ¶ 47. A POSA would understand that the HTTPS Client/server application is the application that transfers content from Client A to the server via, for example, an HTTP “post” command. *E.g., id.* ¶ 54 (“For example, referring to FIG. 4, the HTTPS Client/server 326 of the Client A 102 can transmit an HTTP post

command containing the Client X Content 410 to the Server 106.”).

511. Client A may “**automatically**” transfer a “**copy of a first electronic file**” using the “**second application**” (HTTPS Client/server) that posts content from the client to the server. For example, the sequence in Figure 6 is described as “showing an illustrative method for sharing and synchronizing content using [the] mixed peer-to-peer and server architecture of FIG. 4” *Id.* ¶ 61. An exemplary “**copy of a first electronic file**” disclosed by Sigurdsson as residing on Client A is a “very large” “maintenance manual.” *Id.* ¶ 58. When a user edits a file (e.g., the maintenance manual), the file may be automatically transferred from the second application to the server if another client that needs to receive the updated file is offline. *Id.* ¶ 62 (“The system 400 can enable . . . synchronization using . . . ‘drop box’ techniques. . . . [I]n order for the system 400 to supply the content to an offline client, ***the server can be used as a drop box for the content until the offline client comes online again.***”).

512. Paragraphs 63 and 68 in Sigurdsson explain that content may be sent directly from Client A to Server 106 when a user “edit[s]” or “sav[es]” a document (e.g., a maintenance manual). Sigurdsson, ¶¶ 63 (“***Processing can start*** in step 602 ***when an event occurs***. This step 602 may be similar to the step 202 of FIG. 2, where ***an event is generated in response to an action performed by a user, such as, saving***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

a file or document, accessing a web page, or opening a file or application. *For example, referring to FIG. 4, a user at the Client A 102 may edit and save a document on his work computer.*”), 68 (Client A “*sends the data directly to the server*, as shown by the dashed line pointing from the step 606 to the step 610”). In addition to sending the file when it is saved, it would have been obvious to send the maintenance manual to Server 106 “*when content is changed* on the Client A 102.” *Id.* ¶ 84 (“*The Client A 102 posts Client A Content 108 to the Server 106 when content is changed on the Client A 102*. Posting of the Client A Content 108 occurs, for example, whenever the user saves a file or document [P]osting can occur as soon as an event is generated by an action, such as saving a file. For example, a client can maintain an open connection with a server and post as soon as an event is generated.”).

513. As one way to implement this feature, Sigurdsson describes that an event listener listens for events such as, e.g., when a user modifies a file, and, in response to such events, the Client/server application will send the modified files to the Server. Sigurdsson, ¶¶ 32-33 (“For example, . . . a user at the Client A 102 may edit and save his resume on his home computer. This action may generate an event that is detected by an event listener. . . . For example, after the event triggered by an action is detected, the content from the file associated with the action can be

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

extracted. The extracted content can be the text or images included in the file that was saved, accessed or **modified.**”); *id.* ¶ 45 (“The Client A 102 posts the Client A Content 108 to the Server 106 **after content is changed** on the Client A 102.”); *id.* ¶ 47 (“The **Event Listener 322 listens for and detects user actions**, such as saving a file, deleting a file, or viewing a web page, **that may require the associated . . . Client A Content 108 to be posted to the Server 106.**”). Thus, Server 106 (“**first electronic device**”) may “**receive**” “**a copy of a first electronic file**” (maintenance manual) “**from a second application**” (HTTPS Client/server) “**executable on the second electronic device**” (Client A). It would have been obvious to apply these teachings of automatically sending content from Client A to Server 106 when the content is changed to the exemplary “**first electronic file**” (maintenance manual) when it is modified, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file (the modified maintenance manual is automatically received by Server 106 from the HTTPS Client/server 326 at Client A responsive to the user modifying the maintenance manual).

514. The content transferred to the Server may include a full copy of the file modified by the user. *Id.* ¶ 65 (“In step 606, the content originating on the first client is posted [to the Server]. . . . **The content can include a copy of the document that**

the user saves”); *see also id.* ¶ 8 (“[E]xtracting the content of the first file may include *generating a copy of the first file that retains the first file’s original file formatting.*”); *id.* ¶ 28 (“The Client A Content 108, which can include *a copy of the user’s file*, is sent automatically to the Server 106.”).

515. As a result of Client A posting the content to the server, “[t]he Client A Content 108 is received by and stored in the Server 106.” *Id.* ¶ 85. For example, Sigurdsson discloses that its server may store a modified first electronic copy to a memory device associated with the server. *Id.* ¶ 48 (“The **Server 106 can contain a Storage 330**, a Synchronization Manager 332, and an Authentication Manager 334. The **Storage 330** can contain the Content A, and Content B through Content N 336 that has been received from the user’s various clients.”); *id.* ¶ 74 (“the content may be stored at the server”). Sigurdsson further discloses: “The Server 106 serves as a **temporary storage location** that can facilitate synchronization between the user’s clients.” *Id.* ¶ 83; *see also id.* ¶ 85 (“The Client A Content 108 is received by and **stored in the Server 106.**”). Sigurdsson further discloses “System 900” in Figure 9 and explains that “**the System 900 may be included in** either or all of the Client A 102, the Client B 104, and **the Server 106.**” *Id.* ¶ 104. In connection with the embodiment in Figure 9, Sigurdsson discloses the use of a memory device: “Storage devices suitable for tangibly embodying computer program instructions and data

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

include *all forms of non-volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.*” *Id.* ¶ 110.

516. Sigurdsson further teaches that Server 106 **receives a copy of a modified first electronic file** from a client device, such as Client A. Sigurdsson, ¶¶ 28 (“The Client A Content 108, *which can include a copy of the user’s file*, is sent automatically to the Server 106.”), 54 (“In step 506, the content originating on the client is posted to the server. *The content can include a copy of the document (or the content of the document) that was saved by the user . . .*”), 65 (discussing copy of file saved by user), 85 (same), 103; *see also id.* ¶ 8 (“[E]xtracting the content of the first file may include generating a copy of the first file that retains the first file’s original file formatting.”). I note that although some embodiments refer to peer-to-peer sending of files (e.g., from Client A to Client B), Sigurdsson describes that files may be sent directly to the server system by a client, including in the embodiment described in Figures 4-7. For example, Client A may “**send[] the data directly to the server**, as shown by the dashed line pointing from the step 606 to the step 610. For example, referring to FIG. 4, the *Client X Content 410 can be sent directly to the Server 106 . . .*” *Id.* ¶ 68; *see also id.* ¶¶ 84-85 (describing that Client

A posts content to the Server).

- e. [1c]: “wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and”

517. For the reasons discussed above in connection with element [1b], and the reasons below, Sigurdsson discloses wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file.

518. “Client C” in Figure 4 is an exemplary “**third electronic device**” that includes “**a third application executable on the third electronic device.**” Sigurdsson’s various “clients” are fungible, so it would have been obvious to apply the functionality in this limitation to Clients A, B, or C. See Sigurdsson, ¶ 51 (“The Server 106 can still serve as a drop-box for temporarily (or permanently) storing Client X Content 410 originating from *any the* [sic] *Clients 102, 104 or 406. The drop-box, or storage, capabilities of the Server 106 can be used, for example, if a particular client is offline when the first attempt is made to deliver new content to it.* Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the Client C 406 can have communicat[ion] with the Server 106 in a

manner similar to the method used by the Clients A or B.”). Sigurdsson explains that “Clients A and B 102, 104 can contain similar components and can accept and transmit content in a similar manner. For example, both clients can include Files 346, HTML Files 348, an Index 350, an Event Listener 352, a File to Text/HTML Converter 354 and an HTTPS Client/server 356.” *Id.*, ¶ 50. Because Clients A, B, and C are described as operating similarly, they all could post/push data/files to the Server in the same way using the same components. *See also* Fig. 3 (depicting both Client A and Client B as having identical event listeners and HTTPS client/servers). In addition, the functionality related to transferring data from Client A to the Server and then from the Server to other clients is described generically as applying to any combination of the clients and the server. *See id.* ¶ 68 (“For example, referring to FIG. 4, the Client X Content 410 can be sent directly to the Server 106 The Server 106 can then transmit the content to the remaining client peers, such as the Clients B and C.”). Thus, the functionality for synchronizing files related to any particular client described in Sigurdsson could apply to any other client as well (the clients are fungible).

519. Thus, because Client C could include the same components and implement the same functionality as Client A, it would have been obvious to implement the functionality performed by Client A in claim [1b] with respect to the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

copy of the modified maintenance manual (“**copy of a first electronic file**”) but instead using the HTTPS Client/server (“**third application**”) at Client C (“**third electronic device**”) to transfer a “**copy of a second electronic file.**” For example, an exemplary “**copy of a second electronic file**” at Client C is described as, e.g., “a small office memo” in Sigurdsson. Sigurdsson ¶ 58 (“[T]he user may go to work and update a small office memo on the Client C 406.”).

- f. [1d]: “wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and”

520. Sigurdsson in combination with Shappell discloses wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user.

521. Sigurdsson discloses or renders obvious that the **first application** (Synchronization Manager 332) is configured to “**automatically transfer the modified first electronic file copy**” (copy of the modified maintenance manual) to a “**third electronic device**” (Client C). As discussed above for claim [1b], Client C

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

may communicate with Server 106 using the same HTTPS Client/server used by Clients A and B. Sigurdsson, ¶ 51 (“Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the *Client C 406 can have communicate* [sic] *with the Server 106 in a manner similar to the method used by the Clients A or B.*”). Sigurdsson explains in connection with the embodiment in Figure 6 that Server 106 may send the modified maintenance manual received from Client A to Client C. *Id.* ¶ 80 (“In step 626, the server sends the missing content to the third client. For example, referring to FIG. 4, the *Server 106 sends the Client X Content 410 to the Client C 406.*”).

522. It would have been obvious to use Server 106 to automatically send the **modified first electronic file copy** responsive to receiving the file copy from the second electronic device. Sigurdsson’s Server in Figure 4 includes a “Synchronization Manager” that “synchronize[s] content stored on and shared between” different clients (e.g., “Client A” and “Client C”). Sigurdsson, ¶ 48 (“The Synchronization Manager 332 can use the Time Stamps 338 to synchronize content stored on and shared between the Client A 102 and the Client B 104.”). A POSA would understand that “synchronization” in this context means “version comparisons of copies of . . . files to ensure they contain the same data.” EX1021, Microsoft Computer Dictionary (5th ed. 2002), 506 (entry for “synchronization”);

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

see also id. (“In handheld computing, the process of updating or backing up the data on a handheld computer to the linked software applications on a desktop computer.

Data changes made on the desktop computer may also be copied to the handheld during synchronization.”). Thus, a POSA would have understood and found it obvious that Sigurdsson’s reference to “synchroniz[ing]” content stored on and shared between clients would have encompassed using the Synchronization Manager to automatically send, via the first application, the modified first electronic file copy to a third electronic device responsive to receiving the modified first electronic file copy from the second electronic device.

523. Sigurdsson explains that Client A content received at the Server could be automatically transferred by the Server to Client B, C, or any other client. *Id.* ¶ 68 (“For example, referring to FIG. 4, the Client X Content 410 can be sent directly to the Server 106 ***The Server 106 can then transmit the content to the remaining client peers, such as the Clients B and C.***”). As an example, in connection with the embodiment depicted in Figure 6, Sigurdsson describes an automated process for (1) detecting when Client C comes back online, (2) passing a list containing metadata to enable Client C to see what content it is missing, (3) receiving a request from Client C for missing data, and (4) sending the missing data to Client C. The transfer of “Client X Content 410” could be sent automatically

when a temporarily offline client comes back online. *Id.* ¶ 69. “In step 614, the server determines the content missing from the third client.” *Id.* ¶ 70. “In step 616, ***the server passes the list of missing content to the third client . . .***, which had been offline.” *Id.* ¶ 73.

524. Sigurdsson also discloses sending the ***entire*** file once it has been modified. Sigurdsson, ¶ 8 (“Additionally, extracting the content of the first file may include ***generating a copy of the first file that retains the first file’s original file formatting.***”); *id.* ¶ 28 (“The Client A Content 108, which can include ***a copy of the user’s file***, is sent automatically to the Server 106.”); *id.* ¶ 54 (“In step 506, the content originating on the client is posted to the server. The content can include ***a copy of the document*** (or the content of the document) that was saved by the user as well as metadata that describes information, such as the type of data, its source, when it was saved, and when it was posted to the server.”); *id.* ¶ 65 (“The content can include ***a copy of the document*** that the user saves as well as metadata that describes the type of data, the client that generated the data, and when the data was last changed.”).

525. Sigurdsson’s discussion of synchronization would have motivated a POSA to consider solutions to the problem of a user wanting to keep an up-to-date version of a file on their home and work computers and synchronizing both

computers to make sure they are updated as soon as possible with the most recent version. For example, Sigurdsson expressly discloses automatically transferring a modified file from one client to the server (as discussed above in connection with claim [1b]), and other embodiments describe automatically transferring the modified file from the server to another client when the latter client connects to the server. Sigurdsson, ¶ 31 (“When the Client A 102 (e.g., the user’s home computer) connects to the Server 106, Client B Content 112 originating from the Client B 104 is automatically copied to the Client A 102, as indicated by arrow 116. . . . Similarly, whenever the Client B 104 (e.g., the user’s work computer) connects to the Server 106, any missing Client A Content 108 is copied automatically to the Client B 104, as indicated by arrow 118.”). Sigurdsson further discloses in another embodiment that “a client can maintain an open connection with a server and post as soon as an event is generated.” *Id.* ¶ 84. It would have been obvious in view of these disclosures to automatically transfer a modified file received from Client A to Server 106, and then automatically transfer the modified file, via an “open connection,” to Client C as soon as the file is received at the server.

526. Sigurdsson discloses that a “**first electronic file**” may be “**stored on the third electronic device.**” Sigurdsson, ¶ 81 (“In step 628, the content received by the third client is stored on the client’s local storage device. For example,

referring to FIG. 4, ***the Client X Content 410 received by the Client C 406 is stored on the client's local storage device.***”). Sigurdsson discloses “System 900” in Figure 9 and explains that “***the System 900 may be included in*** either or all of the ***Client A 102, the Client B 104,*** and the Server 106.” *Id.* ¶ 104. It would have been obvious that similar storage could have been included in Client C, given that Clients A, B, and C are described in Sigurdsson as being fungible for the reasons discussed above in connection with claim [1c]. In addition, it would have been obvious that, instead of being a mobile device as shown in Figure 4 that may have potentially had limited memory in November 2008, Client C could have also been a computer with memory large enough to store a large file, such as an updated maintenance manual, for the reasons discussed above in connection with claim [1c] (regarding the various client devices being fungible). *Id.* ¶ 45 (“The Client A 102 and the Client B 104 each represent a computer used by a user including, for example, a personal computer at home, a computers at work, and portable computers, such as a laptop computer, a PDA, and a cell phone.”). As noted above, Sigurdsson explains that its clients are fungible in that they contain the same components and may implement the same functionality with respect to sending and receiving files from the server. *See, e.g., id.*, ¶¶ 50-51. Thus, it would have been obvious to a POSA that the functionality described with respect to Client A could be implemented in Client C and vice versa.

Id.

527. In addition, or in the alternative, it would have been obvious to practice this limitation in view of Sigurdsson in combination with Shappell. Shappell expressly discloses automatically transferring a modified file to another client after receiving the modified file from an originating client. For example, Shappell discloses an automatic replication feature that automatically copies files added to a group's "shared space" to the user's computer "when they are added to the shared space." Shappell, ¶ 81 ("When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file. . . . [I]f automatic replication of the shared file is enabled, the local machine obtains an updated version of the file***, as is shown in step 2006 in FIG. 20."). It would have been obvious to apply this concept from Shappell to Sigurdsson's systems and methods to automatically send a modified file from the server to Client C when it has arrived at the server.

528. Sigurdsson discloses concerns about the limited memory on Server 106 and the need to manage the amount of data stored on Server 106. For example, Synchronization Manager 332 includes "a Space Quota 342 and a Throughput

Threshold 344” that are used to control the storage of data to the server. Sigurdsson,

¶ 48. A POSA would have been motivated in view of these concerns to automatically transfer an incoming, updated file from Server 106 to any other client device that may need the file as quickly as possible to avoid unnecessary use of Server 106’s memory if transfer of the file is delayed. It would have been obvious to combine Sigurdsson with Shappell for the reasons discussed above in connection with ’823 Ground 1, claim [1c].

529. Transferring the second file to Client A would occur automatically, with no user intervention. Moreover, given the fungibility of Sigurdsson’s clients, it would have been obvious to receive a file from Client A at the Server and then automatically transfer the file to Client C in view of the embodiment described in connection with Figure 5, which describes a similar data flow between Clients A and B. Sigurdsson, ¶¶ 52-55 (“In step 508, the server stores the content received from the originating client, and the content is propagated to the second client.”).

530. Sigurdsson’s discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home and work computers and synchronizing both computers to make sure they are updated with the most recent updates on each respective version. For example, Sigurdsson expressly references a client’s work computer and home computer as

two exemplary “clients,” and describes example use cases such as a user editing a Word document that is synchronized between the user’s home and work computers: “For example, a user may edit a Word document file at his home computer. When the user saves the document, the document’s content is automatically extracted and transmitted to the user’s computer at work. When the user searches for the modified document later at his work computer, the content is presented to the user even though the Word document was not previously stored on the user’s work computer.” Sigurdsson, ¶¶ 26-28 (also describing Server 106 as optional “drop box” “conduit” between client devices).

531. A POSA would have found it obvious that in this type of use case, after the copy of the Word document (which may be called “Version 1”) is transmitted from the user’s home computer to the user’s work computer, the user may continue to edit that document on the work computer while the user is at work. Based on Sigurdsson’s bidirectional file synchronization architecture as illustrated in Figure 4, a copy of the updated, edited file at the work computer (call it “Version 2”) would then be transmitted from the work computer, through the Server, and back to the user’s home computer. However, Sigurdsson does not provide details about what would happen at the user’s home computer in that circumstance where “Version 2” of the file is received at the home computer that previously contained “Version 1”

of the same file. A POSA would have been motivated to look for solutions to replace old versions of files with new ones as described in this limitation. A POSA would have found it obvious that one possibility would be for the updated modified file (“Version 2”) to replace the outdated older version of the file (“Version 1”) on the home computer. A POSA would have found it obvious to replace the older version of the file for reasons including the obvious benefit of preventing copies of outdated versions from endlessly accumulating in storage on the user’s computers. Furthermore, Shappell expressly teaches and motivates this feature with its “automatic replication” option, as discussed further below.

532. Shappell’s “automatic replication” option further renders obvious implementing Sigurdsson’s Server 106 **to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user**. Shappell teaches a user-selectable “File Replication” setting where the user can switch between (1) manually downloading files “on an on-demand basis,” and (2) automatically downloading files when available, with an “automatic replication” setting. Shappell, ¶¶ 62, 81. When “automatic replication” is enabled, Shappell teaches automatically replacing an “out-of-date file” with an “updated version” on a receiving computer. *Id.*, ¶ 81 (“When a modification of a file contained in a shared space occurs, a

notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file***. This will change the visual representation of the file to a ‘missing’ file within the shared space folder when the application setting does not automatically update files, as is shown in step 2010 in FIG. 20. On the other hand, ***if automatic replication of the shared file is enabled, the local machine obtains an updated version of the file***, as is shown in step 2006 in FIG. 20.”), Figs. 19-20.

533. It would have been obvious to implement Shappell’s “automatic replication” feature as an option for each Sigurdsson client device, including Client C (the “**third electronic device**”), so that Server 106 would **automatically** transfer to Client C the copy of a file modified at Client A (the “**second electronic device**”), to ensure that the user at Client C would have the most recently updated file. A POSA would have been motivated to incorporate Shappell’s auto-replication to address Sigurdsson’s concern for each client “to keep its locally stored content synchronized with content from the other clients,” and it easily and obviously would be implemented applying Sigurdsson’s teaching that each client can “maintain an open connection” with the server. Sigurdsson, ¶¶ 84-85.

- g. **[1e]: “automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;”**

534. For the reasons discussed above in connection with element [1d], and the reasons below, Sigurdsson in combination with Shappell discloses automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user.

535. For the reasons discussed above in connection with claim [1c], Sigurdsson’s various “clients” are fungible. Thus, for the reasons discussed above in connection with claims [1c] and [1d], it would have been obvious to automatically transfer the modified “**second electronic file copy**” (small office memo) from the server, via the server’s “**first application**,” to Client A (“**second electronic device**”) “**to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user.**”

- h. [1f]: “wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.”**

536. Sigurdsson discloses wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.

537. Sigurdsson explains that “Client A 102 posts the Client A Content 108 to the Server 106 after content is changed on the Client A 102. ***Posting of the Client A’s Content 108 occurs, for example, whenever the user saves a file or document,*** or when he views a webpage.” Sigurdsson, ¶ 45. More specifically, Client A’s Event Listener “listens for and detects user actions, such as ***saving a file*** . . . that may require the associated content Client A Content 108 to be posted to the Server 106.” *Id.* ¶ 47. The updated file may then be sent to the Server (i.e., the “**first electronic device**”) via an HTTP POST command. *Id.* ¶ 54 (“***The content can include a copy of the document (or the content of the document) that was saved by the user*** as well as metadata that describes information, such as the type of data, its source, when it was saved, and when it was posted to the server. For example, referring to FIG. 4, ***the HTTPS Client/server 326 of the Client A 102 can transmit***

an HTTP post command containing the Client X Content 410 to the Server 106.”);
see also id. ¶ 65 (similar). Thus, it would have been obvious to practice this
limitation in view of Sigurdsson.

2. Claim 3

538. Claim 3 recites: “**The system of claim 1 wherein the first application is configured to store the modified first electronic file copy to a memory device associated with the first electronic device.**” It would have been obvious to practice this claim in view of Sigurdsson.

539. Sigurdsson discloses that its Server 106 is **configured to store, via the first application** (Synchronization Manager 332) **the modified first electronic file copy** (modified copy of the maintenance manual) **to a memory device associated with the first electronic device** (Storage 330). Sigurdsson, ¶ 48 (“The *Server 106* *can contain a Storage 330*, a Synchronization Manager 332, and an Authentication Manager 334. The *Storage 330* can contain the Content A, and Content B through Content N 336 that has been received from the user’s various clients.”); *id.* ¶ 74 (“the content may be stored at the server”). Sigurdsson further discloses: “The Server 106 serves as a *temporary storage location* that can facilitate synchronization between the user’s clients.” *Id.* ¶ 83; *see also id.* ¶ 85 (“The Client A Content 108 is received by and *stored in the Server 106.*”). Sigurdsson also expressly discloses “System

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

900” in Figure 9 and explains that “***the System 900 may be included in*** either or all of the Client A 102, the Client B 104, and ***the Server 106.***” *Id.* ¶ 104. In connection with the embodiment in Figure 9, Sigurdsson discloses the use of a memory device: “Storage devices suitable for tangibly embodying computer program instructions and data include ***all forms of non-volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM and DVD-ROM disks.***” *Id.* ¶ 110.

540. As discussed above, the “**first application**” includes Server 106’s “Synchronization Manager 332,” which includes “a Space Quota 342 and a Throughput Threshold 344” that are used to control the storage of data to the server. Sigurdsson, ¶ 48. Thus, a POSA would have found it to be obvious that the Synchronization Manager 332 (“**first application**”) manages not only the sending and receiving of files to and from clients but also the storage of such files to Server 106, and that storage to Server 106 would therefore be “**via the first application**” (Synchronization Manager 332). Also, as discussed above, this functionality could be included as part of an “application server” that could be used to service requests to store data to the server. *Id.* ¶ 112. A POSA would understand that an application server could be used to service requests, such as storing data from clients to the

Server.

3. Claim 4

541. Claim 4 recites: “**The system of claim 3, wherein the first application is configured to store the modified first electronic file copy to the memory device associated with the first electronic device when the third electronic device is not in communication with the first electronic device.**” It would have been obvious to practice this claim in view of Sigurdsson.

542. As discussed above in connection with claim 3, Sigurdsson’s first application is configured to store the modified first electronic file copy to the memory device associated with the first electronic device.

543. In connection with the embodiment depicted in Figures 4-7, Sigurdsson describes that such storage may occur when the **third electronic device** (e.g., Client C) **is not in communication with the first electronic device**. Sigurdsson, ¶ 66 (“In some implementations, the second client would propagate the data directly to a third client if the third client (e.g., the Client C) was online, but in this case, *the third client is offline, so the content is transmitted to the server for temporary storage.*”); *see also id.* ¶ 68 (“An alternative embodiment of step 606 bypasses the second client and sends the data directly to the server, as shown by the dashed line pointing from the step 606 to the step 610.”). Sigurdsson explains that Server 106 acts as a “drop-

box” if an intended recipient, such as Client C, is offline: “The **drop-box**, or storage, capabilities **of the Server 106 can be used**, for example, **if a particular client is offline[.]**” *Id.* ¶ 51; *id.* ¶ 62 (“[I]n order for the system 400 to supply the content to an offline client, the server can be used as a drop box for the content until the offline client comes online again”).

4. Independent Claim 8

544. Claim 8, for purposes of my analysis, is substantially similar to claim 1 except for the preamble, which recites: “A method implementable in an electronic system having a storage component, the electronic system being coupled to a first electronic device having stored thereon a first electronic file and a second electronic device having stored thereon a second electronic file, the first electronic device and the second electronic device each being associated with a user.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 8 (method)
A system comprising: a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:	A method implementable in an electronic system having a storage component, the electronic system being coupled to a first electronic device having stored thereon a first electronic file and a second electronic device having stored thereon a second electronic file, the first electronic device and the second electronic device each being associated with a user, the method comprising:

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and	receiving from the first electronic device a copy of a first electronic file modified by the user, the copy of the modified first electronic file being automatically provided to the electronic system by the first electronic device when a content of the first electronic file is modified by the user;
wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and	receiving from the second electronic device a copy of a second electronic file modified by the user, the copy of the modified second electronic file being automatically provided to the electronic system by the second electronic device when a content of the second electronic file is modified by the user;
wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and	automatically transferring the modified first electronic file copy to the second electronic device to replace an older version of the first electronic file stored on the second electronic device with the modified first electronic file copy having the content modified by the user; and
automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;	automatically transferring the modified second electronic file copy to the first electronic device to replace an older version of the second electronic file stored on the first electronic device with the modified second electronic file copy having the content modified by the user;
wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that	wherein the first electronic device automatically transfers the copy of the modified first electronic file to the electronic system upon determining that

a save operation has been performed on the modified first electronic file.	a save operation has been performed on the modified first electronic file.

545. For the reasons discussed above in connection with claim 1, and the reasons below, claim 8 would have been obvious.

546. Sigurdsson “Client A” in Figure 4 may be considered a “**first electronic device**” and “Client C” may be considered a “**second electronic device**,” and both are in communication with Server 106 (“**electronic system**”), as shown in Figure 4. See Sigurdsson, ¶ 51 (“Although FIG. 4 does not show explicit arrows between the Client C 406 and the Server 106, the *Client C 406 can have communicate* [sic] *with the Server 106 in a manner similar to the method used by the Clients A or B.*”). As discussed above in connection with claim 3, the Server includes a “storage component.” *Id.* ¶ 48 (“The *Server 106 can contain a Storage 330*, a Synchronization Manager 332, and an Authentication Manager 334. The *Storage 330* can contain the Content A, and Content B through Content N 336 that has been received from the user’s various clients.”).

547. Client A is described as, for example, a “user’s home computer” and Client C is depicted as a user’s mobile device. Sigurdsson, ¶¶ 28-29, Fig. 4. In addition, it would have been obvious that, instead of being a mobile device as shown in Figure 4 that may have potentially had limited memory in November 2008, Client

C could have also been a computer with memory large enough to store a large file, such as an updated maintenance manual, for the reasons discussed above in connection with claim [1c] (regarding the various client devices being fungible). *Id.* ¶ 45 (“The Client A 102 and the Client B 104 each represent a computer used by a user including, for example, a personal computer at home, a computers at work, and portable computers, such as a laptop computer, a PDA, and a cell phone.”). As noted above, Sigurdsson explains that its clients are fungible in that they contain the same components and may implement the same functionality with respect to sending and receiving files from the server. *See, e.g., id.*, ¶¶ 50-51. Thus, it would have been obvious to a POSA that the functionality described with respect to Client A could be implemented in Client C and vice versa. *Id.* For the reasons above in claim [1a], each of Clients A, B, and C and Server 106 are “**associated with a user.**”

548. Client A may store a first electronic file. Sigurdsson, ¶ 32 (“a user at the Client A 102 may edit and save his resume on his home computer”). Client C may likewise store a second electronic file on its local storage device. *Id.* ¶ 81 (“Client X Content 410 received by the Client C 406 is stored on the client’s local storage device.”); *see also id.* ¶ 42 (“Client B 104 stores the Client A Content 108 on a local storage device, such as a hard drive in a personal computer.”). As noted above, because Clients A and C are described as operating similarly, both could store

data and post/push data/files to the Server in the same way using the same components. Thus, Client A and Client C could store respective “**first**” and “**second**” electronic files based on their similar configurations and functionality.

5. Claim 10

549. This claim would have been obvious for the reasons below and those discussed in connection with claims 3 and 8. Claim 10 recites: “The method of claim 8, wherein the electronic system is configured to store the modified first electronic file copy to the storage component.” As discussed above in connection with claim 8, Sigurdsson’s Server includes a storage component, and the Server is configured to store the modified first electronic file copy to the storage component for the reasons discussed above in connection with claims 3 and 8.

6. Claim 11

550. This claim would have been obvious for the reasons below and those discussed in connection with claims 4, 8, and 10. Claim 11 recites: “The method of claim 10 wherein the electronic system is configured to store the modified first electronic file copy to the storage component when the second electronic device is not in communication with the electronic system.” As discussed above in connection with claim 8, Sigurdsson’s Server includes a storage component, and the Server is configured to store the modified first electronic file copy to the storage

component when the second electronic device is not in communication with the electronic system for the reasons discussed above in connection with claims 4, 8, and 10.

L. '561 Ground 2: Claims 1, 3, 4, 8, 10 and 11 of the '561 Patent Would Have Been Obvious in View of Sigurdsson in Combination with Shappell and Moromisato.

1. Independent Claim 1

551. Claim 1 of the '561 patent would have been obvious for the reasons discuss above in Ground 1. In addition, it would have been obvious to combine Sigurdsson with Shappell and Moromisato and, in view of these references, claim 1 would have been obvious.

b. [1d]: “wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user”

552. Moromisato discloses a server-based system in which a change to the server data automatically causes the server to send a command to all clients to modify their data: “Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends

information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.***” Moromisato, 1:47-54, 14:47-55.

553. Moromisato also teaches the automatic transfer of the modified file copy to another device to replace an older version of the file stored on the other device with the modified copy that includes the content modified by the user. For example, Moromisato teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system ‘sites’. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the

changes. ***If a file has been edited, then the document share engine may*** either ***disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

554. It would have been obvious to incorporate this functionality into Sigurdsson’s system in view of the motivation to keep a user’s work and home computers (as well as the computers of other team members, as taught in Moromisato) synchronized automatically as changes are made to avoid the situation in which people view/edit files that are not up-to-date.

555. A POSA would have been motivated to look to references like Moromisato, which also discloses “synchronizer” functionality—specifically, functionality designed to operating in conjunction with the Windows operating system.¹⁹ Moromisato, 7:9-24 (“The file system RAMP program 304 is part of the collaborative system code and detects changes to files in the OS file system 302. It communicates with the ***file synchronizer 308***, as shown schematically by arrow 340,

¹⁹ Moromisato, assigned to Microsoft, describes functionality related to the then- and now-popular Microsoft Windows operating system. A POSA—and almost everyone else using computers at the time—would have been familiar with the basic file system concepts described in Moromisato, and a POSA would have been motivated to look at Microsoft patents and patent publications at the time as an industry leader in this technical field.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

in order to inform the file synchronizer 308 that it needs to synchronize those changes. In particular, ***the Windows XP Pro operating system provides a system event notification system to which clients can subscribe to receive notifications of selected events. These events include changes in the OS file system, such as folder and file edits***, additions and deletions. . . . During the collaborative system start up, the file system RAMP program 304 is invoked so that it can register with the OS event notification system for file notifications indicating ***changes to files*** and file contents in the Windows shell file system 302.”).

556. Like Sigurdsson, Moromisato describes embodiments that use a “server based” system to implement file synchronization functionality. Moromisato, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. ***Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.***”). A POSA considering Sigurdsson’s similar server-based embodiments would have been motivated to consider the design of Moromisato’s file synchronizer illustrated in Figure 3:

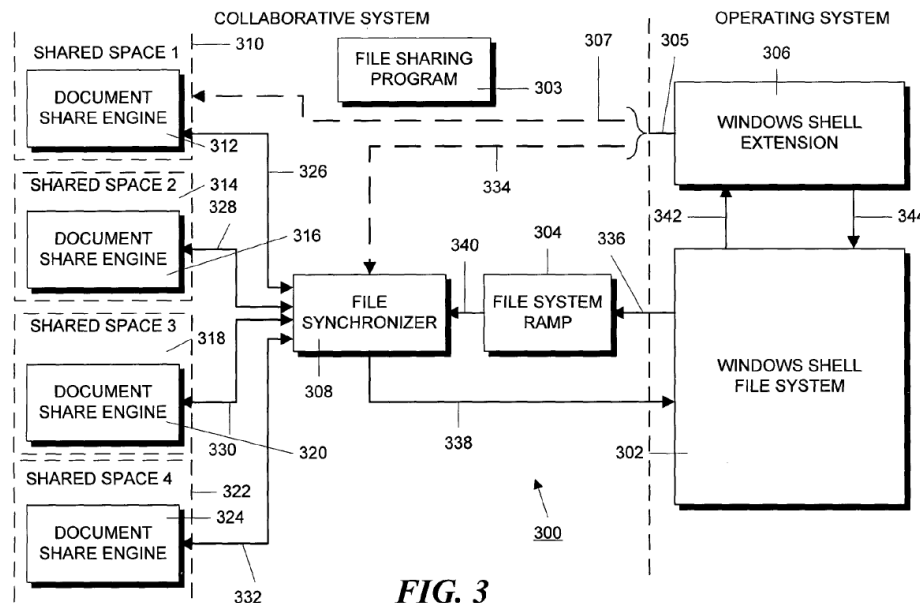


FIG. 3

Id., Fig. 3; *see also id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system and vice versa.”).

557. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified file copy to a user device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44 (“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

replaces an older version of the file stored on the device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***”); see also *id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing).

558. In view of these teachings from Moromisato, it would have been obvious to use the Synchronization Manager in Sigurdsson’s system to automatically transfer a copy of a file to a client device associated with the user to replace an older version of the file stored on the client device. And it would have been obvious to combine Sigurdsson with Moromisato and Shappell for the reasons discussed above in connection with ’823 Ground 2.

- c. **[1e]: “automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the**

**modified second electronic file copy having the content
modified by the user”**

559. Because, as discussed in Ground 1, Sigurdsson’s clients are fungible and implement the same functionality, and for the reasons above in connection with element [1d], this claim limitation would have been obvious.

2. Claims 3, 4, 8, 10 and 11

560. For the reasons discussed above for claim 1, and the reasons discussed above in the corresponding sections of Ground 1, claims 3, 4, 8, 10 and 11 would have been obvious in view of Sigurdsson in combination with Shappell and Moromisato.

**VII. THE CHALLENGED CLAIMS OF THE ’823, ’787, ’607, ’622, ’942, AND ’561
PATENTS WOULD HAVE BEEN OBVIOUS IN VIEW OF BROWN IN
COMBINATION WITH HESSELINK AND ONE OR MORE OF RAO AND/OR
SHAPPELL**

A. Summary of the Prior Art

1. Brown [EX1025]

561. Brown discloses a system that includes a server and distributed client devices that “learn about changes made to the files on the server” and “push local changes of the files onto the server.” Brown at Abstract. Brown further discloses a “synchronization application” that is “used to synchronize the clients and server, synchronizing metadata and selected files.” *Id.* More specifically, a user’s “[c]lient machines synchronize with each other by synchronizing to a common server

account” in embodiments disclosed in Brown. *Id.*, 2:65-67. “From the user’s perspective, the synchronization process is automatic, runs in the background and requires minimal monitoring or intervention.” *Id.*, 3:40-42.

562. The client devices described in Brown monitor their respective file systems for “changes to send to the server SFS [application] during synchronization.” Brown, 3:47-50. The server SFS (“Synchronization File System”) organizes files and directories into folders, which are assigned to respective users. *Id.*, 2:59-62, 3:62-66. Users may log in to access the files and directories. *Id.*, 3:66-4:2. On any given user’s account, the server “enforces a single changer model in which only one client at a time can change the state of the user’s server data. When multiple clients of a single user account push changes to the server concurrently, the server SFS interleaves the changes.” *Id.*, 4:59-64.

563. Each change to a file or directory is assigned “a sequence number called the sync (short for synchronization) index.” Brown, 5:5-7. “Synchronizing a client machine with server data that is not up to date with its server account can be viewed as the process of moving the state of the client’s directory from the old server SFS state, identified by an older (lower) server sync index (SSI) value, to a new server SFS state, identified by a more recent (higher) SSI value.” *Id.*, 5:7-13. A client initiates the synchronization process by passing its “client sync index (CSI)” to the

server, “identifying [the client’s] current known state of the account.” *Id.*, 5:14-17.

If the client is already up to date, the server returns “the SSI (having the same value as the CSI) back to the client” to indicate that the client is up to date. *Id.*, 5:19-21.

“Otherwise, the server SFS returns the new higher SSI along with the server metadata information the client needs to transition its account from its current state to the server’s current state.” *Id.*, 5:21-24.

564. The server stores a “three-level hierarchy of sync indices (SIs) in its metadata. Brown, 5:25-26. The SSI is at the highest level. *Id.*, 5:26-28. The next level is the “directory sync index (DSI).” *Id.*, 5:31-32. “The server SFS maintains a directory table for each user account with a directory item for each user directory. Each directory item contains a DSI value associated with the last change to a file or directory within the directory. The server SFS uses this value to quickly find directories with changes that need to be pushed down to a syncing client.” *Id.*, 5:32-38. At the lowest level, each file and folder is assigned a “file sync index (FSI)” that identifies file versions. *Id.*, 5:41-49. Similar metadata is stored on client devices and is compared with server metadata to determine when and which files and directories need to be updated on the client devices. *See, e.g., id.*, 6:60-7:1, 7:21-43 (listing similar client-side metadata fields).

565. Brown’s client devices monitor their local file systems for activity.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Brown, 7:3-4. “When file system activity occurs, the client SA [“Synchronization Application”] records the event in the client metadata.” *Id.*, 7:4-5. Brown refers to embodiments of the portion of the client SA that monitors the client’s file system as a “filter driver.” *Id.*, 7:11-13. “During synchronization, when the client SA pulls down changes from the server and makes changes to the user’s directory, it updates the client metadata to reflect those changes.” *Id.*, 7:13-16. “Also, when the client SA pushes changes to the server during the second part of the synchronization process, it records new SID [“Server ID”] and FSI values returned by the server SFS into the client metadata file and directory items.” *Id.*, 7:16-20. “[F]ilter driver 350 watches for other applications accessing the files in folder 302, so as to determine which files on the client have been changed. When the client later synchronizes with the server, the client can use the information provided by the filter driver to identify which files to push to the server.” *Id.*, 8:15-20.

566. In Brown’s embodiments, metadata is sent by the client devices before any actual file data is sent. *See, e.g.*, Brown, 10:7-11. The metadata sent by a client is compared to server metadata associated with a user’s files to determine which files and directories need to be synchronized. *Id.*, 10:12-42. When a client uploads a new file to the server, the server sends back the SID associated with the file. *Id.*, 10:55-60. The client can also notify the server when a file stored on the client has been

renamed, to instruct the server to update its version of the file. *Id.*, 10:60-62. Brown describes two main options for uploading changes to files.

567. When uploading a file from a client to the server, Brown explains that the file may be divided up into “blocks” and, if fewer than a certain number of the blocks have changed, then only those blocks of the file that have changed will be uploaded to the server. Brown, 12:36-49. On the other hand, if the number of blocks that have changed exceeds a threshold limit, then the entire file is uploaded to the server. *Id.*, 12:49-51. A similar process may be carried out when determining to download an entire file or only part(s) of the file from the server. *Id.*, 13:5-20.

568. Brown explains that, in some embodiments, filter drivers on clients can track changes made locally, “even while disconnected from the server.” Brown, 17:64-66. Such changes can then be migrated to the server with the client devices reconnect to the server. *Id.*, 17:66-18:1.

2. Hesselink [EX1026]

569. Hesselink discloses a synchronization system in which a user may have multiple computers that synchronize versions of a given file, such as a home computer, work computer, and laptop computer: “The present system may also be optionally employed to perform synchronization of files among a user’s multiple computers. Referring back to the operations described above, the system is designed

to automatically update at least the ‘master copy’ of a file at a remote location (i.e., the file at the location of the device module from where the file was accessed by the user module) as the user makes changes to the file locally on the computer associated with the user module.” Hesselink, ¶ 210. “Thus, the user is ensured of having a completely up to date master copy of each shared data file at the home location/home locations (since home locations of shared files may vary). For example, the system may be set up such that when a user is away, working on his/her laptop, a personal file is automatically updated at the user’s home computer storage device, while a business file is automatically updated at the user’s work computer storage device.” *Id.* ¶ 192.

570. Hesselink teaches that when the user modifies or edits a file, the system automatically transmits the changes. The synchronization occurs “as soon as the update has been performed” at one device (for example, a home or office location), so that all connected computers “may be automatically updated concurrently with the updates of the device module file, each time an update is performed” Hesselink, ¶¶ 183, 189. “As the user works on the file, such as by adding to the file, modifying or editing, each time that the user saves his/her work, the system compares the differences between the saved version of the file and the previous version of the file, and, as long as the user module remains connected to the network,

the changes/differences are transmitted over the network to the remote device module, which are saved there, thereby maintaining the home file (i.e., official reference file, source file, master file or DataClient file) up to date with the latest revisions, as described above.” *Id.* ¶ 189. “Optionally, all computers that are connected to the network, that share a file that the user is working on and that have a copy of that file in cache memory (or on a remote device) may be automatically updated concurrently with the updates of the device module file, each time an update is performed, when such computers are actively subscribed, as noted above. Computers that are not actively subscribed or are offline can be automatically updated when reconnecting with the network or initiating an active subscription.” *Id.* Hesselink reiterates: “As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. Computers that are offline at such a time will be automatically updated as soon as those computers go online again.” *Id.* ¶ 183; *see also id.* ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user

module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”).

571. Hesselink discloses “automatically and repeatedly” checking whether synchronizing devices are in communication prior to initiating synchronization: “As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.*** The persistence of the system ensures that updates are completed for all online computers, as if there is some failure or problem during the update, the connection restores itself to ensure that the update is completed.” Hesselink, ¶ 183; *id.* ¶ 190 (“With this functionality, when the user works on a file, it is automatically updated in the remote device by the device module. Once the file has been put into the cache memory on the local computer, the user can work locally on the file via the user module, as described above. Therefore if the connection to the network is lost, this will not affect the user’s ability to work on the file. Updates to the official reference file (and optionally the file as

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

located in other remote computers) obviously cannot be performed as long as the connection is broken. Furthermore, ***the system is persistent and may be configured on a user computer to automatically and repeatedly attempt to reconnect with any and all remote devices which are actively subscribed to when a connection is broken during an active subscription.*** The present software recognizes that the file is in cache and the user can continue working locally when the connection is broken. When the network connection is restored, the software automatically updates the home file (official reference file) on the remote device via the local user module's transmission to the remote device module, as described above (and optionally, on any other computers where the file may reside). This feature is useful not only for when a network connection is accidentally broken, but in situations where the user moves to a location where the user knows a network connection is not possible, as in the example described above, while flying on an airplane, or in some other location that is not network accessible. Even in these situations, ***as soon as the user logs into the network again, the official reference file may be automatically updated,*** so that when the user returns to the location of the home file (official reference file), the home file is already up to date.”).

572. Hesselink describes indicating the availability of files using metadata and discusses the importance of distinguishing metadata from file data more

generally: “Targets of the data access requests that are made can generally be classified into two main categories: file overhead information [i.e., metadata] and file data. File overhead information pertains to file attributes or properties that may include but are not limited to file name, file size, folder status, last modified date, created date, etc. Every file or folder has file overhead information that can be used by local applications to get properties information about a file or a folder. For example, Windows Explorer® may use file overhead information to list or display directory data structures for user navigation purposes. On the other hand, file data is the actual binary content of a file. For example, the file data of a digital photograph contains the actual jpeg binary data that can be used for graphical display and/or printing.” Hesselink, ¶ 134.

573. Hesselink further describes: “Once permissions have been set up for particular files and/or desktop and/or devices, and for particular users, and fully granted as described, when a user module connects with the network, icons and/or text representative of files that are accessible from remote devices may be integrated into the file structure of the computer associated with that user module. Thus, the ***files, folder, directories, etc., that are accessible from remote devices appear automatically on the display of the computer associated with that user module.*** For example, FIG. 14B shows user interface 1400 wherein the ‘Shared With Me’

function has been selected, such as by clicking on or otherwise selecting the tool button 1412.” Hesselink, ¶ 167. “If the local computer is running Microsoft Windows®, then once the user module of the local computer connects with the network for the first time after access has been granted and thereafter until access is withdrawn, then the user can go to ‘My Computer’ and see that ***the file integration feature displays an icon and/or other indicator or text for each of the folders, files, drives, directories, etc. that permission has been granted for.*** Thus, in this example, in addition to icons for the local user’s directories, etc., icons for Remote 1—Music Folder, Remote 2—C Drive, Remote 3—Video Folder, and Remote 4—Video Folder are displayed, since these have been integrated into the user’s file structure, and they appear along with the C drive, A drive, and any other accessible drives, directories or files that are displayed under the user’s file structure (e.g., ‘My Computer’). Similar file structure integration functionality exists with other operating systems, although the display location may not be labeled ‘My Computer.’” *Id.* ¶ 187. “Thus the present invention functions as if the user is storing the file on the user’s local device (e.g., local hard drive), but it is actually functioning as a distributed drive, since the local file is only being stored in cache, and the file storage is occurring in the remote device by the device module.” *Id.* ¶ 191.

3. Rao [EX1019]

574. See ¶¶ 136-140, above.

4. Shappell [EX1020]

575. See ¶¶ 141-144, above.

**B. '823 Ground 3: Claims 1-4, 8-11, and 13-16 of the '823 Patent
Would Have Been Obvious in View of Brown in Combination with
Hesselink, Shappell, and Rao.**

1. Independent Claim 1

576. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with
computer program instructions that, when executed, cause the server system
to:

[1b] receive, over a network, a copy of a first file from a first client device
associated with a user, wherein the copy of the first file is automatically
received from the first client device responsive to the user modifying a content
of the first file stored on the first client device, the copy of the first file being
an updated version of the first file that is generated from the user modifying
the content of the first file;

[1c] receive, over a network, from the first client device, first metadata
associated with the updated version of the first file that is generated from the
user modifying the content of the first file, the first metadata being assigned a
first priority greater than a second priority assigned to the copy of the first file;

[1d] automatically transfer, based on the first priority being greater than the
second priority, the first metadata over a network to a second client device
associated with the user such that the first metadata is transferred to the second
client device before the copy of the first file is transferred to the second client
device,

[1e] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and

[1f] (ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and

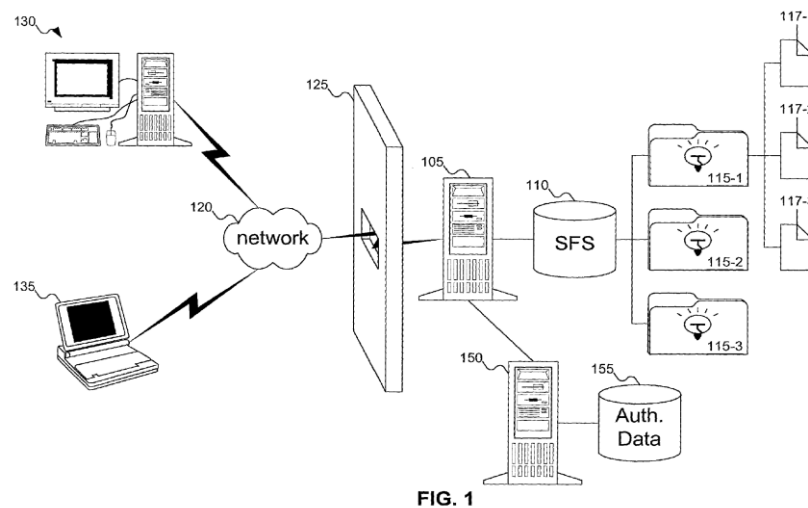
[1g] wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and

[1h] subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,

[1i] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

b. [1p]: “A system:”

577. The preamble of claim 1 recites “[a] system.” I understand that the preamble of a claim generally does not limit the claim, but to the extent that the preamble provides a claim limitation, it is disclosed by Brown.



578. Brown discloses a client/server **system** shown in Figure 1, including at least two client devices, such as desktop computer **130** and portable computer **135**. “Client **130** represents an ordinary desktop computer, as might be present at a user’s place of work or home. Portable computer **135** represents a notebook or laptop computer, as a user might take with him to a hotel room on business.” Brown, 4:20-24. The client devices communicate with server **105**.

579. Brown describes “a client/server application that allows users to synchronize data on multiple machines. The server component is a server Synchronization File System (SFS) with a functional interface and metadata organization that enables efficient and reliable synchronization by the clients. . . . The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server.” Brown, 2:57-65.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Brown contemplates a system with “multiple clients of a single user account”

Id., 4:62.

580. For these reasons and the reasons that follow, Brown renders the preamble obvious.

- c. **[1a]: “a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:”**

581. Brown discloses and renders obvious a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server to perform operations.

582. In the system in Figure 1, the “**server system**” is server **105**. The “server **105** includes server SFS **110**.” Brown, 3:56-57. “The server component is a server Synchronization File System (SFS) with a functional interface and metadata organization that enables efficient and reliable synchronization by the clients.” Brown, 2:59-62. “Typically, server SFS **110** is installed on a hard disk drive within server **105**” *Id.*, 3:59-60.

583. A POSA would have understood that Brown’s server includes or obviously would include **one or more processors programmed with computer program instructions that are executed to cause operations** given that Brown discloses that “[s]erver 105 includes all the typical elements of a server, such as a

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

central **processor**, **memory**, bus, disk space, etc.” Brown, 3:57-59. Claims 38 and 39 of Brown further specify media containing a “program for a synchronization file system of a server” to synchronize a directory on the file system with a directory on a client, including a number of “software” elements at the server. Brown, claims 38-39. A POSA would have understood and found it obvious that in order to achieve the functions of the server system in Brown including those performed by SFS 110, as discussed further below in connection with the claim elements, that one or more processors in the server system obviously would be programmed with computer program instructions that would cause the programmed functions to occur, given that this would have been routine and standard operation for a typical server including “all the typical elements of a server” as Brown describes. *Id.*, 3:57-59.

- d. **[1b]: “receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file”**

584. Brown in combination with Hesselink disclose and render obvious receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the

first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file.

585. For purposes of my analysis, the “**first client device associated with a user**” is disclosed by the client devices such as client **130** in Brown. Brown discloses and renders obvious multiple client devices that are each associated with the same user. Brown addresses the problem in the prior art that an individual user may have multiple computers, such as work, home, and portable computers, and “needs to access the same files from the various computers”:

But as computers have become more affordable, people find themselves working with several machines. It is increasingly common for people to find themselves working on one machine at the office, a second machine at home, and having use of a portable computer when they need to have computer access while traveling. . . .

But with the increasing number of machines a person might find himself using comes an added complexity. Since a person typically accesses the same files from the various computers, the user needs to be certain that the files he is accessing are current.

Originally, people carried files on floppy disks from one machine to the next. But the increases in file size sometimes make floppy disks impractical. And if the user forgets to bring the files with him as he moves around, or forgets to move the latest versions of the files off the computer he most recently used, the user can find himself with several versions of the files, each of which contain desired portions.

Accordingly, a need remains for a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made, to address these and other problems associated with the prior art.

Brown, 1:24-50. Consistent with these teachings, Figure 1 specifically discloses two client computers, client **130** which is a user's desktop computer, and client **135** which is the user's portable computer: "Client **130** represents an ordinary desktop computer, as might be present at a user's place of work or home. Portable computer **135** represents a notebook or laptop computer, as a user might take with him to a hotel room on business." *Id.*, 4:20-24. In view of Brown's teachings regarding the problem to be solved, a POSA would have understood that computers **130** and **135** are both, and obviously would both be, associated with the same user. Brown contemplates "multiple clients of a single user account" that may "push changes to the server concurrently" and that "the server SFS interleaves the changes." *Id.*, 4:62-64. Therefore, "each user can have several clients accessing a single folder." *Id.*, 11:55-56. Thus, Brown discloses and renders obvious multiple client devices associated with a single user.

586. Brown describes the synchronization of the client/server system of Figure 1 in Figures 4A-4C and 5 with reference to client **130**. Brown, Figs. 4A-4C, 5, 10:7-11:52. Brown further illustrates the synchronization of the client/server system of Figure 1 by describing detailed algorithms for client devices to synchronize with the server, including uploading and downloading metadata and files, in the flowcharts shown in Figures 8A-11F and corresponding text. *Id.*, Figs.

8A-11F, 13:57-16:39. Brown teaches that these flowcharts specifically correspond to the clients and server shown in Figure 1, namely client devices **130** and **135** and server **105**. *Id.*, 13:58-61 (“synchronizing the clients and server of FIG. 1”), 14:7-9, 14:60-62, 15:30-32. A POSA thus would have understood from Brown’s disclosures that the disclosed synchronization process applies to changes to a file that are made on either of client devices **130** or **135** associated with the same user, which will then be synchronized to the other client device through the server. A POSA would have understood that the reference to the client devices associated with a user (such as devices **130** and **135**) are fungible in the sense that files would be synchronized bi-directionally between client devices **130** and **135** as files are modified on either of the devices.

587. Brown’s server 105 is also disclosed as being associated with that same user, at least because the server SFS stores folders that are each associated with each user and can only be accessed by a given user once that user is logged in. Brown, 3:62-4:2 (“Stored on server SFS 110 are folders 115–1, 115–2, and 115–3 Each of the folders is assigned to a user. Once the user has logged in, the user can access the files and directories (cumulatively called directory entries) within the folder.”). Each user has a server “account.” *See id.*, 4:62-64 (one user account may be associated with multiple client devices). “Client machines synchronize with each

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

other by synchronizing to a common server account.” *Id.*, 2:65-67. Changes to files are synchronized to “a user’s server account.” *Id.*, 5:3-4.

588. Brown discloses or at least suggests that its server **105** is configured to **receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file.**

589. Brown discloses and renders obvious that the **user modifies a content of an electronic file stored on the client device**, creating an “updated” version of the file. “Often, when files are **updated**, only a portion of a file is changed. For example, when a text document is edited, some paragraphs are removed, and other paragraphs are inserted.” Brown, 11:64-66.

590. Brown discloses and renders obvious that, responsive to the modification, and as part of the file synchronization process disclosed in Brown, the server receives a copy of the updated/modified file. “**When the client is uploading a file to the server, the client starts by making a copy of the file.** The client SA uses the filter driver to read the file. The filter driver makes sure that the copy operation

does not interfere with an application attempting to access the file during the copy. Copying the file is relatively quick, and ***once the copy is made the client SA can operate on the copy of the file*** without worrying about another application on the client trying to access the file.” Brown, 11:22-30.

591. Brown discloses that server 105 receives **a copy of the file** when more than a threshold portion of the file has changed. Brown teaches that either only the parts of the file that have changed or the entire file may be transferred, depending on the extent of the file changes. Specifically, Brown discloses that the “entire” modified file is uploaded to the server from the client device when more than a threshold portion of the file has changed. The threshold is used by separating the file into blocks and performing a hash function on each block and determining message digest array (MDA) values. “MDAs are arrays of 16-byte message digests computed from each 4K block of a file.” Brown, 12:15-16. “Hash function **605** takes a block of the file, such as block **610** of file **615**, and computes the message digest, such as message digest **620** in MDA **625**. MDA **620** can then be used to determine if the file can be only partially uploaded.” *Id.*, 12:41-45. Then, “if less than a threshold number of message digests in the MDAs match, ***the entire file is transmitted.***” *Id.*, 12:49-51; *see also id.*, Fig. 11B (steps 1117, 1122, 1120), 15:41-54 (describing that “if a previous version of the file is on the server,” and if

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

“an insufficient number of message digests match between the received and computed MDAs,” then at step 1120 the client “uploads the entire file”).

592. Brown discloses “automatic” synchronization of modified files. “From the user’s perspective, ***the synchronization process is automatic***, runs in the background and requires minimal monitoring or intervention. In an embodiment of the invention, the client SA initiates synchronization on a fixed time interval. But a person skilled in the art will recognize that ***the client SA can use any scheduling algorithm to initiate synchronization***. In addition, the user can also initiate synchronization at any time manually.” Brown, 3:40-47.

593. As noted, in “an embodiment” of Brown, “the client SA initiates synchronization on a fixed time interval,” but alternatively the system “can use any scheduling algorithm to initiate synchronization,” and the user may manually initiate synchronization “at any time.” Brown, 3:40-46. Once synchronization is initiated, a file that has been modified on a client device will be transferred to the server (where the entire file is transferred) as a result of that modification, as discussed previously and shown in Brown’s flow chart figures. To the extent Brown were argued not to disclose the first client automatically transferring a copy of a file “responsive to the user modifying a content” of the file, **Hesselink** discloses and renders obvious that feature.

594. It would have been obvious in view of Brown in combination with Hesselink for the server described in Brown to automatically receive a copy of a modified file when the user modifies a content of the file.

595. Brown teaches that the Synchronization Application (“SA”) running on each of its client devices may monitor activity in the local client file system to identify changes to files. “The client SA monitors local file system activity within the directory (sometimes called a folder) on the client so that it can efficiently locate changes to send to the server SFS during synchronization.” Brown, 3:47-50. In particular, “filter driver **350** is used to monitor the activity of files within the folder on the client. Specifically, filter driver **350** watches for other applications accessing the files in folder **302**, so as to determine which files on the client have been changed.” *Id.*, 8:13-17.

596. It would have been obvious to implement the Brown system so that the server receives a copy of the modified file when a user modifies the file on a client device, rather than waiting for a fixed time interval. Brown teaches that the client Synchronization Application “can use any scheduling algorithm to initiate synchronization” and that the user can also initiate synchronization “at any time.” Brown, 3:44-47. Brown also seeks to provide “***a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are***

made” *Id.*, 1:46-50. A POSA would have found it obvious to look to Hesselink for a technique to serve Brown’s stated purpose to distribute files automatically across multiple clients to maintain currency at each client as changes are made.

597. Hesselink teaches a synchronization system where a given individual user may have multiple computers that synchronize versions of a given file, such as a home computer, work computer, and laptop computer, similar to Brown’s disclosure that each user has multiple client devices for which files are synchronized. “The present system may also be optionally employed to **perform synchronization of files among a user’s multiple computers**. Referring back to the operations described above, the system is designed to **automatically update** at least the ‘master copy’ of a file at a remote location (i.e., the file at the location of the device module from where the file was accessed by the user module) **as the user makes changes to the file locally** on the computer associated with the user module.” Hesselink, ¶ 210. “Thus, the user is ensured of having a completely up to date master copy of each shared data file at the home location/home locations (since home locations of shared files may vary). For example, the system may be set up such that when a user is away, working on his/her laptop, a personal file is automatically updated at the user’s home computer storage device, while a business file is automatically updated at the user’s work computer storage device.” *Id.* ¶ 192.

598. Hesselink teaches that when the user modifies or edits a file, the system automatically transmits the changes to the file over the network. The synchronization occurs “as soon as the update has been performed” at one device (for example, a home or office location), so that all connected computers “may be automatically updated concurrently with the updates of the device module file, each time an update is performed” Hesselink, ¶¶ 183, 189. “***As the user works on the file***, such as by adding to the file, ***modifying or editing***, each time that the user saves his/her work, the system compares the differences between the saved version of the file and the previous version of the file, and, as long as the user module remains connected to the network, ***the changes/differences are transmitted over the network to the remote device module***, which are saved there, thereby maintaining the home file (i.e., official reference file, source file, master file or DataClient file) up to date with the latest revisions, as described above.” *Id.* ¶ 189. “Optionally, all computers that are connected to the network, that share a file that the user is working on and that have a copy of that file in cache memory (or on a remote device) ***may be automatically updated concurrently with the updates of the device module file, each time an update is performed***, when such computers are actively subscribed, as noted above. Computers that are not actively subscribed or are offline can be automatically updated when reconnecting with the network or initiating an active

subscription.” *Id.*

599. Hesselink reiterates: “As noted above, ***when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.***” Hesselink, ¶ 183.

600. Hesselink further teaches that its disclosed technique of automatically updating copies of a file among a user’s multiple computers as the user makes changes to the file “may be set to apply to only one file or any number of files” at the user’s option: “The present system may also be optionally employed to perform synchronization of files among a user’s multiple computers. Referring back to the operations described above, the system is designed to automatically update at least the ‘master copy’ of a file at a remote location (i.e., the file at the location of the device module from where the file was accessed by the user module) as the user makes changes to the file locally on the computer associated with the user module. However, the user may be provided with alternative synchronization options by the present system. For example, the user may choose to always synchronize all copies of a file, wherein, whenever the user makes a change to that file locally, at any location, then all of the devices which subscribe to that file are automatically updated immediately. ***This option, and all others described below, may be set to apply to***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

only one file or any number of files to which the user module is subscribed. At the device module end, such options may be chosen to apply in the same way. Another synchronization option provided by the present system is one in which a local copy of the file is stored on the local computer, and changes with at least one other remote device are automatically synched.” Hesselink, ¶ 210.

Motivation to Combine Brown with Hesselink

601. It would have been obvious to combine the teachings of Brown and Hesselink, predictably resulting in the synchronization system of Brown in which the scheduling algorithm causes the upload of a modified file from a user device responsive to the user modifying the file as taught by Hesselink.

602. As an initial matter, Brown and Hesselink are analogous references that are both in the same field as the Challenged Patents. Both references are directed to techniques for effectively synchronizing modified versions of files between client devices over a network. Both references are reasonably pertinent to the problems faced by the inventor of the challenged patent for improving upon existing file synchronization systems.

603. The combination would have been a straightforward matter of combining prior art elements according to known techniques to yield predictable results, namely the system of Brown as modified in view of Hesselink’s teachings

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

as summarized above. The modification would have provided a predictable improvement to Brown's system, improving the Brown system in the same way that the relevant features enhance Hesselink's disclosed system, by synchronizing files from a client device to a server responsive to the user modifying the file.

604. In addition, based on the specific teachings of Brown and Hesselink, a POSA would have been motivated to consider Hesselink in view of Brown specifically for the causation of file synchronization. For example, a POSA would have been motivated to consider Hesselink in view of Brown's stated goal to provide "a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made" Brown, 1:46-50. In addition, Brown would have encouraged a POSA to consider alternative scheduling algorithms for synchronization, including those discussed in Hesselink, given that it expressly teaches that "any" algorithm may be used: "In an embodiment of the invention, the client SA initiates synchronization on a fixed time interval. But a person skilled in the art will recognize that ***the client SA can use any scheduling algorithm to initiate synchronization.***" *Id.*, 3:42-46.

605. Hesselink also provides express motivations to combine its teachings with Brown's to implement Hesselink's automatic concurrent transmission of changes to files, including (1) saving time for the user, and (2) avoiding the problem

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

of having two different versions of a file, or an outdated version of a file. Hesselink teaches: “As the user works on the file, such as by adding to the file, modifying or editing, each time that the user saves his/her work, the system compares the differences between the saved version of the file and the previous version of the file, and, as long as the user module remains connected to the network, the changes/differences are transmitted over the network to the remote device module, which are saved there, thereby maintaining the home file (i.e., official reference file, source file, master file or DataClient file) up to date with the latest revisions, as described above. ***This functionality saves the user a great deal of time***, since there is no need to synchronize files when the user returns to the site of the device module. ***Further, there is no concern over having two different versions of a file, or concern that a user is not using the latest, most up to date version of a file.***” Hesselink ¶ 189. These express motivations naturally harmonize with Brown’s stated goal to provide “a way to maintain distributed files across multiple clients, ***maintaining currency at each client as changes are made***” Brown, 1:46-50.

606. In addition, given Brown’s concern with maintaining currency of files across multiple user devices, a POSA would have found it obvious in view of Brown to look to Hesselink’s teaching to automatically push updates to files to connected clients as soon as file updates occur. For example, Hesselink discloses: “As noted

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

above, *when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.* Computers that are offline at such a time will be automatically updated as soon as those computers go online again.” Hesselink, ¶ 183.

607. A POSA would have had at least a reasonable expectation of success in implementing the modified system of Brown in this manner. Brown already discloses that the client can use any scheduling algorithm to initiate synchronization. Initiating synchronization responsive to the user modifying the file would have been well within the grasp of a POSA, as well as implementing the modified system of Brown accordingly, using well-known computer software and networking knowledge and design principles. The references provide ample detail such that a POSA would have been able to implement the modification without undue experimentation.

608. I note that a POSA would not have perceived any obstacle or dissuasion from the fact that Hesselink discloses that in its system, only the changes to a file are transmitted in synchronization, rather than transmitting the entire modified file, which consumes less bandwidth than transferring the entire file. Brown itself teaches that either only the changes, or the entire file, can be transferred, depending on the

extent of the changes to the file (whether below or above a threshold), as discussed previously. A POSA would therefore understand that Brown already contemplates and accounts for a balancing and trade-offs between transferring only the changes and transferring the entire file. For example, a POSA would understand that transferring only changes generally would consume less bandwidth to transfer, but would provide a more complicated approach, requiring tracking of portions of the file that have changed, and would consume more processing time and resources to deal with the differential analysis and reconstitution, whereas transferring the entire file would be a simpler approach. Thus, a POSA would have understood that Brown already accounts for the possibilities of transferring only changes or transferring an entire file, but Brown goes further than Hesselink with more nuanced functionality that will transfer the entire file if a certain amount of the file has changed. In other words, a POSA would have understood that while Hesselink contemplates only a binary “all or nothing” approach where a system would always transfer only the entire file or only a differential, Brown provides a more sophisticated approach in this regard.

609. Furthermore, it is immaterial that Hesselink’s system transfers only changes given that Hesselink’s relevant teaching for purposes of the obviousness inquiry is to synchronize files as changes are made to the files, serving Brown’s

stated goal of “maintaining currency at each client as changes are made,” Brown, 1:46-50, and consistent with Brown’s teaching that any synchronization algorithm can be used to initiate synchronization, *id.*, 3:44-46.

610. In addition, in connection with considering the use of bandwidth to synchronize files, a POSA would have appreciated Hesselink’s teaching that its approach of automatically synchronizing changes to a file “among a user’s multiple computers” “as the user makes changes to the file” “may be set to apply to only one file or any number of files” at the user’s option. Hesselink, ¶ 210. This teaching naturally harmonizes with Brown’s disclosure that files within a certain designated “directory” (or “folder”) on each user client device are synchronized, as well as Brown’s teaching that any synchronization algorithm can be used and that synchronization could also be initiated manually at any time. Brown, 6:61-7:20, 3:44-50. A POSA would have appreciated Hesselink’s teaching and motivation that Brown’s synchronization system could apply different synchronization algorithms to different subsets of files in the monitored synchronized directory in order to differentiate between different needs and values associated with different files or types of files and also account for different bandwidth availabilities (which could provide faster or slower uploads/downloads depending on the user’s network connection, see for example Brown at 12:4 and Hesselink at ¶¶ 3-4 and 12 noting

the availability of high-speed and reliable network connections such as Ethernet at a hotel and DSL, where a “typical Ethernet local area network can facilitate 100 mbps for both upload and download data transfer rate” and noting that “high bandwidth” data networks “can be provided at relatively low cost”). For example, a user with fast and reliable Ethernet network connections for their client devices could designate either their entire synchronized directory, or a subset of a particular time-sensitive or important work file(s) (or subdirectory of files) to be automatically synchronized as changes are made to the file(s) as described in Hesselink, while optionally leaving the remainder of less time-sensitive or less important files in the user’s synchronized directory to be synchronized only on a periodic basis. A POSA applying Hesselink’s teachings to Brown would be motivated to design the system to permit different synchronization algorithms for different subsets of files in this regard.

611. It is similarly immaterial to this combination of teachings that Hesselink generally contemplates a peer-to-peer model while Brown uses a client/server model. Hesselink discusses certain features of prior art systems that Hesselink desired to improve upon, for example, at ¶¶ 8-9 discussing then-current solutions including the “time consuming and tedious responsibility” for a user to synchronize files and “the ability to upload files to a central server” that was

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

“cumbersome and time consuming to use” and ¶ 22 discussing a “virtual drives” system that had its own drawbacks from the user’s perspective. A POSA would have recognized, however, that Brown’s system provides a user-friendly automatic synchronization approach that does not suffer from the perceived drawbacks of those prior art solutions: “From the user’s perspective, the synchronization process is automatic, runs in the background and requires minimal monitoring or intervention.” Brown, 3:40-42. Hesselink also does not specifically disparage or motivate away from applying its relevant synchronization scheduling function to a client/server synchronization system. Notably, Hesselink’s summary of the “need” in the prior art and its “Summary of the Invention” section do not specifically discuss or distinguish the form of network model as between peer-to-peer or client/server. Hesselink, ¶¶ 23-34. A POSA would have considered Hesselink’s relevant teachings to be applicable to improve upon Brown’s user-friendly automatic client/server synchronization systems as well, in view of Hesselink’s express motivation to serve the stated goals of Brown of maintaining currency among a user’s multiple devices, and Brown’s suggestion that any synchronization scheduling algorithm may be used, as discussed previously. Hesselink also does not teach that the devices in its synchronization network must not include any servers. On the contrary, Hesselink specifically teaches that its disclosed system may include servers among the

computers that are synchronized: “As noted above, any of computers **72**, **74** and **76** may be a desktop computer, laptop computer, personal digital assistant, network attached storage system, **file server** or other computing device that has CPU, storage and connectivity capabilities.” Hesselink, ¶ 122. A POSA would have understood that a “file server” in this context would be a server for storing files and making the files available to client devices, much like the server in the Brown client/server system.

- e. **[1c]: “receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file”**

612. Brown in combination with Shappell and Rao discloses and renders obvious receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file.

613. Brown discloses that the server **receives over a network from the first client device** (for example, desktop computer **130** connected to server **105** over network **120** as shown in Figure 1) **metadata associated with the modified file that is generated from the user modifying the content of the file**. Brown describes

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

that various metadata associated with a file is maintained on the client device, including “update” time, file length (that is, the size of the file based on the amount of data it contains), and other information generated from the user modifying file content. “The client SA creates a client Synchronization File System (CSFS) on each client machine to coordinate the synchronization process with the server SFS. This file system contains **metadata** but no file data. Data files reside within the directory on the client as files native to the operating system of the client machine. Like the server metadata, **the client metadata includes file and directory items with fields such as name, update and create time, and file length.**” Brown, 6:61-7:1. Brown further teaches: “The client SA monitors file system activity within the user’s directory on the client. **When file system activity occurs, the client SA records the event in the client metadata.**” *Id.*, 7:3-5; *see also id.*, 7:21-43 (describing the “sync fields in the client metadata” including client ID (CID), Server ID (SID), File Sync Index (FSI), Client change time, and Flags). A POSA would have understood from these disclosures that when the user modifies a file which generates an updated file and the client SA records the event in the client metadata, the updated metadata includes all metadata that was generated by the user updating the file, including the “update” change time and the updated file length/size.

614. Brown renders obvious that the metadata associated with a modified

file is **assigned a greater (first) priority than a lower (second) priority of the file itself** for transmission between the client and server. During the file synchronization process, the client transfers updated metadata to the server. “The client SA now checks its own client metadata for any changes it needs to push to the server. . . . ***On all change operations except for delete, the client SA passes the client change time (adjusted to server time) to the server.***” Brown, 9:40-42, 9:52-54. In one embodiment, the client initiates synchronization by sending metadata in the form of a “CSI” (“client sync index”) to the server. *Id.*, 10:9-10. In another embodiment, the client sends metadata associated with an updated file in the form of an “SID” (“Server ID”), “parent directory SID,” and “the file name” to the server prior to sending the file itself. *Id.*, 15:38-40; *id.*, 4:50-52 (file name is metadata).

615. Brown further discloses: “The client SA synchronizes with the server by ***synchronizing the client metadata with the server metadata.***” Brown, 7:44-45. Brown discusses metadata items on the client device that “need[s] to be pushed to a syncing client” as part of the synchronization process.” *Id.*, 5:37-38. “Metadata **330** shows the metadata for file **320** as stored within CSFS **110 335**, part of client SA **337**. (Although metadata are not shown for the other files and folders within folder **302**, a person skilled in the art will recognize that such metadata exist.) In metadata **330**, file **320** is shown as having a name (which is typically not

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

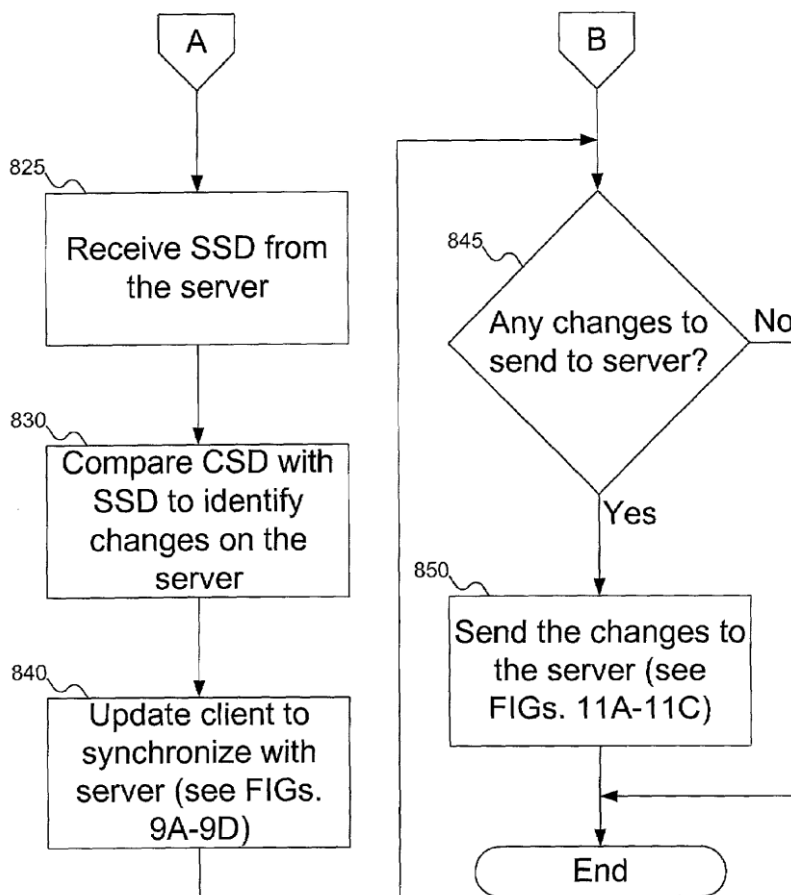
encrypted, although the name can be encrypted in an alternative embodiment of the invention), a CID of 0x 62, a SID of 0x2A, a FSI of 35, the change time of the file, and the flags used in the synchronization process (such as ***identifying metadata items that need to be pushed to the server***). Note that metadata 330 is not shown to store the data of file 320, which is stored in the native operating system of computer 130 within the folder structure, as expected.” *Id.*, 7:58-8:4.

616. In addition, ***before*** transferring a copy of the file from the server to a client (including another client computer that is synchronizing with the server to download updates to files, such as client 135), the server first transfers to that receiving client the Server Synchronization Data (“SSD”) that contains the ***metadata*** of all items, including files, ***“that have been changed”*** since the last synch by the client: “The server SFS maintains a three-level hierarchy of sync indices (SIs) in its metadata. . . . At the lowest level is the SI field that resides in each file and directory metadata item. It records the file sync index (FSI) of the last move or rename for the item (either a file or a directory) or the FSI associated with the creation of the item. The server SFS uses this value to locate ***individual metadata items that need to be sent to polling clients during the synchronization process***. These include any metadata items with a FSI value greater than the CSI value passed in by a client’s sync poll call.” Brown, 5:25-49; 8:48-61 (describing SSD contents).

“Server **105** then sends the SSD to client **130** (in response to a request for the SSD by the client), informing the client of the pertinent changes, as shown in box **415**. Specifically, the SSD includes the SSI, the SIDs of any directories that contain changes since the last time client **130** synchronized with server **105**, the SIDs of all items (files and directories) in the changed directories, and ***the metadata of all items (files and directories) that have been changed*** since the last time client **130** synchronized with server **105**. As mentioned above, by comparing the SSD with the CSD, client **130** can determine what changes have been made to the account on server **105**.” *Id.*, 10:19-30; *see also id.*, Fig. 8B (step 825), 13:65-67 (“If the client is not in sync with the server, then at step 825 (FIG. 8B), the client receives the SSD from the server.”).

617. Accordingly, the client receiving the metadata from the server receives the metadata prior to receiving a copy of the file associated with the metadata. Only after transferring the metadata to the client as part of the SSD (step 825), the client uses the SSD to determine whether the client needs an updated file from the server, and if so, the client requests the file from the server and downloads the file. “As mentioned above, by comparing the SSD with the CSD, client **130** can determine what changes have been made to the account on server **105**. Referring now to FIG. 4B, the four possible results of the comparison of the CSD and SSD are shown. In

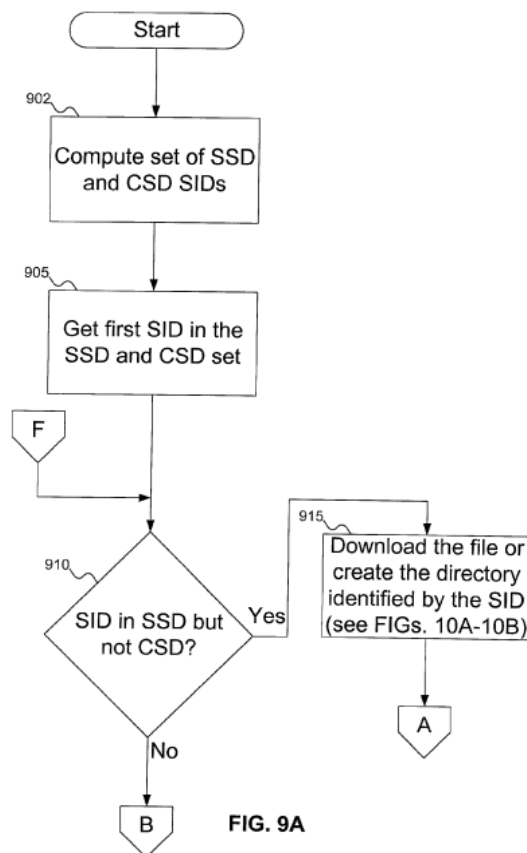
box **420**, a SID is found in the SSD but not the CSD. Client **130** then requests the



appropriate file from server **105** or creates the appropriate directory in the folder on the client.” Brown, 10:28-35.

618. Figures 8B and 9A show the process including (1) at step **825**, the client receives SSD from the server (which includes file metadata), followed by (2) at step **915**, client downloads the file from the server, which is further described in Figures 10A-10B for a modified file, including step **1050** where an entire modified file is downloaded from the server:

619. Brown describes the process: “If the client is not in sync with the server, then at step **825** (FIG. 8B), the client receives the SSD from the server. At step **830**, the client compares the CSD with the SSD to identify any changes on the server that the client is lacking. At step **840**, the client synchronizes with the server to download any changes on the server.” Brown, 13:65-14:3. “FIGS. 9A–9E show a flowchart of the procedure used to pull changes from the server to a client of FIG. 1, according to an embodiment of the invention. In FIG. 9A, at step **902**, the client computes the SSD and CSD set of SIDs, which is the union of the set of SIDs in the directories of the SSD with the set of the SIDs in the same directories of the CSD. At step **905**, the client selects an SID in the SSD and CSD set. At step **910**, the client checks to see if the SID is in the SSD but not the CSD. If the SID is in the SSD but not the CSD, then there is a file or directory on the server not on the client. At step **915**, the client downloads the file from the server or creates a directory.” *Id.*, 14:7-19. As shown in Figure 9A, the process of downloading the file at step **915** is further described in Figures 10A-10B.



620. Figures 10A-10B and their corresponding text describe the process that results in downloading an entire modified file from the server at step **1050**. Brown, Figs. 10A-10B, 14:60-15:16. Brown describes: “If the client was able to locate a previous version of the file, then at step **1025** (FIG. 10B) the client copies the previous version of the file to a temporary file, using the filter driver read function. At step **1030**, the client computes the MDA for the temporary file. At step **1035**, the client retrieves the MDA for the file from the server. At step **1040**, the client compares the received and computed MDAs. At step **1045**, the client checks to see how many message digests in the compared MDAs matched. If an insufficient

number of message digests matched between the compared MDAs, or if the client could not locate a previous version of the file at step **1020** (on FIG. 10A), then at step **1050** the client downloads the entire file.” *Id.*, 15:4-16.

621. Brown therefore describes that, in the synchronization process, metadata generated from modifying the file is transferred with a priority greater than transferring a copy of the modified file itself.

622. Although Brown describes and renders obvious that its synchronization process transfers metadata from the server to the receiving client prior to transferring the associated updated file to that client, Brown does not expressly describe a categorical assignment of priorities to metadata and associated files. To the extent it were argued that Brown does not disclose the limitations of assigning a first, greater priority to metadata and a second, lower priority to files, these limitations would have been obvious in view of Shappell and Rao.

623. Brown in view of Shappell and Rao discloses and renders obvious a predictable modification to Brown’s system so that metadata generate by a user modifying a file is automatically transmitted from the originating client device to the Brown server and then automatically transmitted to other associated and connected client devices through a priority assignment configuration implemented at Brown’s server.

624. A POSA would have recognized a natural and compelling motivation to combine the relevant features of these references to predictably improve upon Brown's synchronization system. In particular, a POSA would have recognized that Brown's disclosed synchronization process leaves open the possibility that the user of one client device (for example, client device **130** as shown in Brown Figure 1) may have modified a certain file but a second client device connected to the network that has an earlier version of the same file (such as client device **135**) will not immediately receive a notification that the file has been modified. That raises the problem that the user at the second client device might want to work with that file, but would not know whether the local copy of the file was up-to-date. Significantly, even where the Brown system has been modified in view of Hesselink so that a client device will transfer a copy of a modified file to the server as changes are made to the file, as described previously, another client device might not receive a copy of the modified file until after a substantial amount of time has passed, such as where a relatively large file is being transferred over a relatively slow network connection. As Brown notes, "if a user has a slow network connection . . . it can be time-consuming to have to wait for the entire document to upload or download." Brown, 12:8-11.

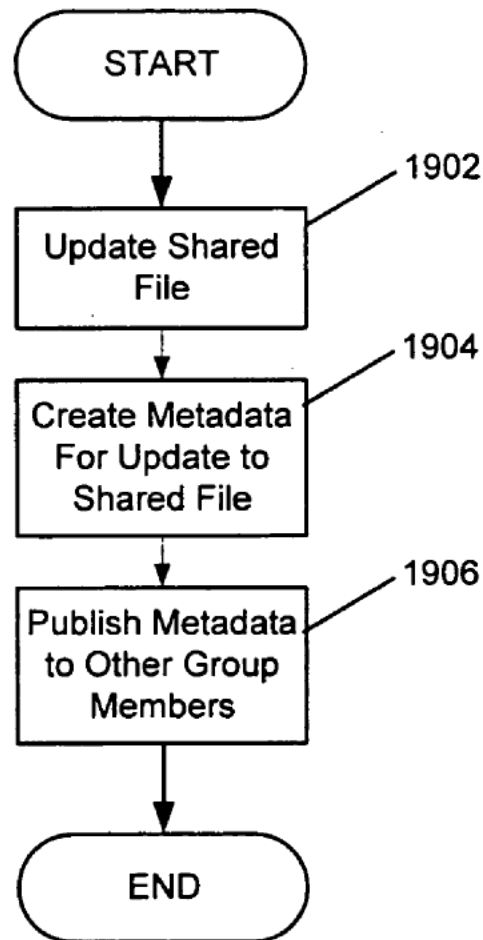
625. A POSA would have recognized that the relevant teachings of Shappell

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

and Rao would address this issue, predictably resulting in the system of Brown where the metadata associated with a file modified at the originating first client device (for example, device **130**) is transmitted automatically by that device to the server, and then a priority assignment configuration implemented at the server causes the sever to transfer the received metadata to a second connected client device for synchronization (for example, device **135**) with a first priority that is higher than a second priority that is assigned to file, thereby enabling a graphical availability indication to be provided at the second client device.

626. **Shappell** and **Rao** both further disclose and motivate this obvious modification to Brown.

627. Like Brown, **Shappell** teaches synchronizing files across multiple computers using metadata. Shappell, ¶ 9. Shappell discloses “shared spaces” in which files may be shared among group members’ computers. *E.g., id.; id.* ¶ 47. Shappell teaches that “[w]hen a **modification of a file**” occurs, “**metadata is transmitted to all members**” of the group. Shappell, ¶¶ 81, 76, Fig. 19 (steps 1902 (“Update Shared File”), 1904 (“Create Metadata For Update to Shared File”), 1906 (“Publish Metadata to Other Group Members”)).



(Shappell, Fig. 19.)

628. Shappell teaches that dissemination of metadata to all group computers serves an important function: “[t]o **inform a user** of a group shared space file that is not present locally” by displaying an indicator at each computer “**to denote a ‘missing’ file that is available on another system.**” Shappell, ¶¶ 79, 76 (the “metadata are sufficient to render an icon of the shared file and may include date, time, thumb-nail information, name size, and optionally the source of the

information”). In one illustration, the indicator is an icon that is “ghosted or semi-transparent.” Shappell, ¶¶ 79, 81.

629. If a version of the file already exists locally, the receipt of metadata causes the computer to either “remove the local copy of the out-of-date file,” which “will change the visual representation of the file to a ‘missing’ file,” or alternatively obtain “an updated version of the file” if “automatic replication” is enabled. Shappell, ¶ 81, Fig. 20 (steps 2002, 2004, 2010, 2006); *id.*, ¶¶ 62 (describing file replication alternatives), 77-78 (describing types of metadata).

630. Thus, Shappell teaches that a file’s metadata—such as file’s “time”—is transferred first (with a higher priority) from an originating computer to all connected group computers **before** transferring a copy of the file itself, to beneficially facilitate user-interface display of the file’s availability.

631. **Rao** provides naturally complementary teachings to categorically prioritize synchronizing metadata before files between server and client, to advantageously enable functions at the client. Like Brown, Rao describes file synchronization between a client and a server over a network. Rao, ¶¶ 26, 37. Rao discloses synchronizing files from a server to a client in two phases: first the metadata, and then the “raw data” of the files themselves. *Id.*, ¶ 63 (“In the **first phase, metadata is synchronized between a server and client** while the raw data

remains on the server system. The client system, when connected or disconnected, may use and manipulate the metadata for any purpose whatsoever, including selecting one or more raw data files that are to be available on the client system. ***The second phase of synchronization is to synchronize the selected raw data to the client system.***”). Much like Shappell, Rao teaches that prioritizing metadata enables functions using the metadata beneficial to “the user of the client” receiving the metadata, such as to “display” information about files. *Id.*, ¶¶ 63 (“the client system . . . may use and manipulate the metadata for any purpose whatsoever”), 43 (“Because the metadata relating to file 140 is contained in file backed item 132, certain functions and operations may be available to the user of the client system 104,” such as “to display . . . audio tracks . . . and perform other operations using only the metadata”).

Motivation to Combine Brown and Hesselink with Shappell and Rao

632. It would have been obvious to combine Brown and Hesselink with the metadata-priority teachings of Shappell and Rao, predictably resulting in Brown’s synchronization system, as modified by Hesselink, where a priority algorithm at the first client and at the server assigns a first, greater priority to metadata and lesser priority to files, resulting in the metadata transferring first (based on the first priority) before transferring associated files.

633. Brown, Hesselink, Shappell, and Rao are analogous references in the same field as the '823, directed to synchronizing files and associated metadata. All three references are directed to techniques for effectively synchronizing modified versions of files between devices over a network. And all three references are reasonably pertinent to the problems faced by the inventor of the challenged patent for improving upon existing file synchronization systems.

634. The combination would have straightforwardly combined prior art elements according to known techniques to yield a predictable improvement to Brown's system, improving the Brown system in the same way that the relevant features enhance Hesselink's, Shappell's, and Rao's disclosed systems, by automatically transmitting metadata as a priority based on file changes to timely notify users that an updated version of a file is available (Shappell) and assigning higher priority to that metadata over a lower priority assigned to files (Rao).

635. A POSA also would have been motivated to consider both Shappell and Rao given their originations with leading file-management vendor Microsoft, as Brown likewise contemplates standard client computers running Microsoft Windows operating systems. Brown, 7:3-11.

636. Furthermore, a POSA would have been motivated to consider additional ways to address Brown's stated bandwidth concerns. Brown, 4:55-57. A

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

POSA would have recognized that Brown's system gives rise to a potential gap in synchronicity between versions of a synchronizing "very large document" over a slow network connection, without notifying the user (as discussed above). Indeed, as Rao cautions, "synchronizing" a very large set of files "may take many minutes or even several hours, depending on the throughput of the communication path . . . among other things." Rao, ¶ 48. As an illustration, the user of a synchronizing client device (e.g., work computer) in Brown may work on modifying the large file. When the user subsequently accesses another synchronizing client device (for example, home computer) to continue work on that file, the user would benefit from being able to quickly and visually determine whether or not the version of the file on the home computer is current, before opening the file to work on it. In view of this potential deficiency in Brown, a POSA would have been motivated to consider Shappell and Rao for a solution that immediately transmits metadata generated by modifying the file to the user's other client devices, advantageously leveraging the metadata to inform a user that an updated file is available. Shappell, ¶ 79 (motivating use of metadata "[t]o **inform a user** of a group shared space file that is not present locally" by displaying an indicator "**to denote a 'missing' file that is available on another system**"); Rao, ¶¶ 63 (motivating that "[t]he client system . . . **may use and manipulate the metadata for any purpose whatsoever**"), 43

(motivating that the metadata enables that “***certain functions and operations may be available to the user***” of the receiving client, including to “display” information regarding files).

637. More generally, the practice of prioritizing the transfer of metadata before its associated content, and the benefits of doing so, would have formed part of the background knowledge of a POSA. By the early 2000s, well-known commercially available software prioritized metadata transfer for similar reasons. For example, as explained in Boyce, Microsoft Outlook downloaded headers for email messages—which comprise metadata such as a message’s subject, date, sender and recipients—before downloading the messages themselves. Boyce, 366-370. Boyce echoes the user interface benefits noted above, explaining that the “advantage” of this functionality “is the ability to work with messages headers of waiting messages without having to download the messages themselves. You can simply connect to the e-mail server, download the headers for new messages, and disconnect. You can then take your time reviewing the message headers to decide which messages to download, which ones to delete without reading, and which ones to leave on the server for later.” Boyce, 366; *see also id.*, 368 (“When you want to process messages selectively”). Boyce provides a relevant example of a large message included among other messages on the email server, explaining that the

functionality to prioritize obtaining email headers “is extremely useful when you’re pressed for time but have a message with a large attachment waiting on the server. You might want to retrieve only your most critical messages without spending the time or connect charges to download that message and its attachment.” *Id.*, 366. “To accomplish this, you can connect with remote mail and select the messages you want to download, leaving the one with the large attachment on the server for a less busy time when you can download it across the network or through a broadband Internet connection.” *Id.*; *see also id.*, 196 (describing functionality to configure system to only download message headers for messages larger than a specified size).

638. As another example of commercially available software, Boyce explains that Microsoft Outlook Express included functionality to send and receive newsgroup messages. Boyce, 307-308. Boyce describes how Outlook Express would first download the headers—information such as message subject, size and date—for newsgroup messages before downloading the messages themselves. *Id.* (“After you select a newsgroup, Outlook Express downloads a certain number of message headers from the newsgroup by default. . . . [Reading a newsgroup message] can be as simple as opening the newsgroup and clicking the message header to download the message and view it in the preview pane.”).

639. A POSA would have readily appreciated that the advantages of

prioritizing metadata transfer that Boyce describes in the context of email and newsgroup messages—as well-known ways of sharing files and other content among users and computer systems—would have similarly applied to the file sharing and synchronization systems described by Brown, Hesselink, Shappell, and Rao.

640. A POSA would have reasonably expected success in implementing this modified Brown system, as Brown already discloses metadata and file transfer using standard, familiar computer, OS, and network components. A POSA would not have been dissuaded by Shappell’s primary embodiment being peer-to-peer rather than client-server. A POSA would have understood that the relevant features of Shappell relating to transfer of metadata and graphical indication of file availability are independent of the type of network architecture as between peer-to-peer or client-server. While Shappell’s exemplary embodiment is “server-less” (e.g., Shappell, Abstract, ¶ 7), Shappell does not specifically disparage a client/server implementation, and proclaims that the “type of network . . . is not critical” and is only “preferably” peer-to-peer, and suggests also transferring “indirectly” between user machines such as would be provided by a client/server architecture (*id.* ¶ 40). Similarly, a POSA would not have been dissuaded by Shappell’s statement that, in its exemplary Figure 2 embodiment, “[e]ach user device will **typically** be used by or associated with a different user” (*id.*, ¶ 40), given the compelling applicability of

Shappell's relevant teaching to improve Brown's system as explained above.

641. A POSA would have been further motivated to look to references like Shappell, given that it was assigned to Microsoft Corporation, in view of the popularity of Microsoft Windows at the time and a POSA's motivation to study the various file sharing and synchronization solutions invented by Microsoft that are disclosed in, e.g., Shappell (which indicates on its cover page that it was assigned to Microsoft). As noted above, a POSA would have known that Microsoft was a significant vendor at the time in file management solutions, including file sharing solutions such as those described in Shappell.

642. A POSA reviewing Brown would have been motivated by Brown's stated goals of maintaining currency among a user's multiple devices and Brown's suggestion that any synchronization scheduling algorithm may be used, as discussed previously, to consider other solutions aimed at addressing the bandwidth concerns addressed in Brown. One of these solutions would have been Shappell's solution of sending metadata associated with files first (before sending the actual file data). A POSA would have known that metadata of a file is often smaller in size than the file itself—particularly when the file is large—and could therefore be transmitted more quickly and with lower bandwidth consumption than the file itself. A POSA would have understood the benefit of looking for solutions like Shappell's to allow users

to see which files, and particularly which modified updated versions of files, are available via metadata before actually downloading the files themselves in their entirety.

643. A POSA would have had at least a reasonable expectation of success in implementing the modified system of Brown in this manner. Brown already discloses the presence of metadata and the use of standard personal computers and operating systems that were very familiar to a POSA. Transmitting metadata and providing a graphical availability indication as taught by Shappell would have been well within the grasp of a POSA, as well as implementing the modified system of Brown accordingly, using well-known computer software and networking knowledge and design principles. The references provide ample detail such that a POSA would have been able to implement the modification without undue experimentation.

644. Hesselink would not deter or dissuade a POSA from further combining the above-discussed features of Brown and Hesselink with the above-discussed features of Shappell and Rao. Hesselink specifically discusses client computers running the Microsoft Windows operating system (Hesselink, ¶¶ 78, 187, 203), a connection server 14 running Windows NT (*id.*, ¶ 80), and the use of the Windows Explorer user interface “to access remote storage (*id.*, ¶¶ 127-128, 133-134). As

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

such, a POSA would have viewed patent references assigned to Microsoft like Shappell and Rao, which discuss Microsoft user interface functionality, to be complementary references, particularly in view of the fact that Hesselink specifically discusses Windows Explorer’s use of “file overhead information” (i.e., metadata “to list or display directory data structures for user navigation purposes.” (*Id.*, ¶ 134.) A POSA would have been motivated to look to references like Shappell and Rao for more detail on such user interface features. In addition, Hesselink discusses separate “[f]ile subscription[s]” for metadata (“file overhead subscription”) and file data (“file data subscription”)—i.e., subscribers could choose to prioritize synchronization of only metadata—and a POSA would have been motivated to look to references like Shappell and Rao to understand the benefits of prioritizing transfer of metadata over file data to facilitate graphical indications of file availability. (*Id.*, ¶ 136.)

645. These numerous commonalities in subject matter between Brown, Hesselink, Shappell, and Rao would have motivated a POSA to consider the best aspects of each reference, as detailed above, and a POSA would have been motivated to implement them as obvious design choices.

- f. **[1d]: “automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device”**

646. Brown in combination with Shappell and Rao discloses and renders obvious automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device.

647. As discussed in connection with element [1c] above, Brown discloses that as part of the synchronization process, the server automatically transfers file metadata to the synchronizing client device prior to transferring a copy of the file associated with the metadata. Furthermore, as discussed for element [1c] above, it would have been obvious to implement Brown’s system in view of Shappell and Rao so that metadata received by the server from the first client device would be automatically transferred, based on a first priority assigned to metadata by a priority assignment configuration at the server, to the second client device before the copy of the first file is transferred to the second client device. Thus, for the reasons discussed above in connection with element [1c], it would have been obvious to send

first metadata from a client to the server to indicate the modification of a file and, subsequently, for the server to communicate that same metadata to another subsequently synchronizing client prior to sending the updated file itself.

- g. [1e] and [1f]: “wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and (ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device”**

648. Brown in combination with Shappell discloses and renders obvious wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata.

649. As discussed in connection with element [1c] above, Brown discloses that as part of the synchronization process, the server automatically transfers file metadata to the synchronizing client device prior to transferring a copy of the file associated with the metadata. Thus, for the reasons discussed above in connection with element [1c], it would have been obvious to send first metadata from a client to the server to indicate the modification of a file and, subsequently, for the server to

communicate that same metadata to another subsequently synchronizing client prior to sending the updated file itself.

650. Brown and Hesselink disclose that their clients may run operating systems that a POSA would have known to include graphic user interfaces, such as Microsoft Windows. *See, e.g.*, Brown, 7:6-11; Hesselink, ¶ 187. In view of these disclosures, a POSA would have found it obvious to consider how to indicate to users the current state of files being synchronized using Brown and Hesselink systems and methods and would have been familiar with Microsoft Windows as a well-known operating system including graphical user interface components for indicating the state of file transfers and sharing, as disclosed in Shappell, for example. Thus, a POSA would have looked to references like Shappell for insight into how to indicate to users the state of files before, during, and after synchronization via graphical user interfaces.

651. Shappell discloses a “**graphical availability indication**” that, “**before the copy of the first file is transferred to the second client device,**” is “**presented at the second client device based on the first metadata.**” As discussed for [1c], Shappell teaches that a computer’s receipt of metadata caused by a file modification causes the display of a “ghosted or semi-transparent” icon. Shappell, ¶¶ 79, 81, Figs. 19-20. The ghosted icon discloses one example of the claimed “**graphical**

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

availability indication,” as it graphically indicates that an “updated version of the file” is available. *Id.*, ¶¶ 79, 81.

652. As explained for [1c], Shappell renders obvious using the “**first metadata**” described in the foregoing limitations as originating from a client described in Brown to cause the presentation of Shappell’s exemplary “**graphical availability indication**” at a “**second client device**” based on the “**transfer of the first metadata to the second client device**,” predictably improving Brown’s system with a convenient and user-friendly notification displayed at the second client device to inform the user of the second client device that an updated file is available from the server. Shappell, ¶¶ 76, 79, 81.

653. Shappell further renders obvious that the “**graphical availability indication**” “**is presented proximate a file icon representing the first file on a user interface of the second client device**” (claim [1f]). Beyond the exemplary “**graphical availability indication**” (the “ghosted” semi-transparent icon) discussed above for [1e], Shappell renders obvious alternative availability indications that obviously would be presented **proximate a file icon representing the first file**.

654. For context, Shappell teaches the use of the well-known Microsoft Windows OS and motivates that its shared space “is presented through a graphical user interface with the look and feel of existing file system features,” such as

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

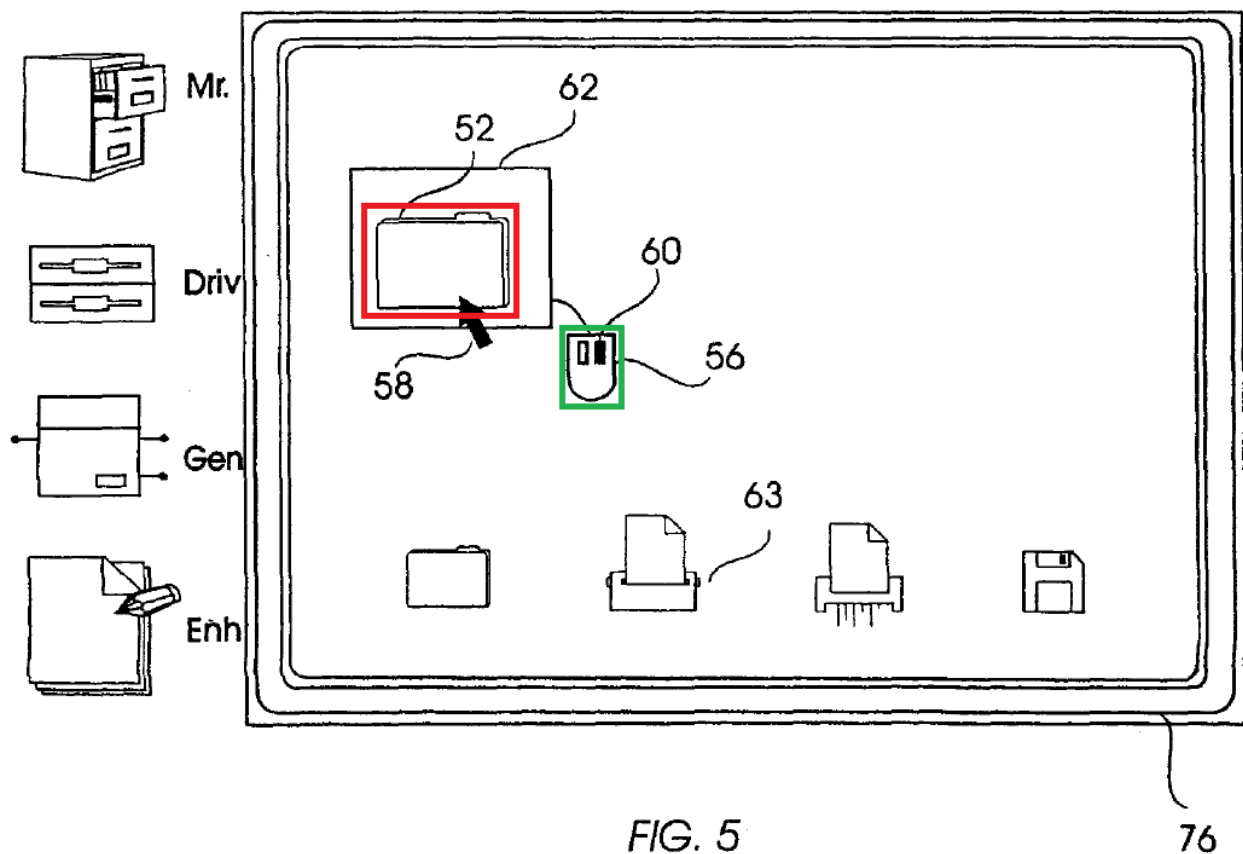
“Windows Explorer-style dialog boxes,” with “**graphical user interface presentations and controls that are included as part of an existing file system, or at least an intuitive extension of existing file system operations.**” Shappell, ¶¶ 42, 45, 47, Fig. 3 (“Windows Core OS” layer 302).

655. Windows provided a plethora of user-interface features, including numerous graphical displays of information about a file presented **proximate a file icon**. Shappell itself teaches several examples, and a POSA would have understood in view of Shappell’s stated GUI goals that those are merely examples among many GUI design-choice options, both available in Windows and “intuitive” extensions.

656. For example, Shappell describes that to “enable the user to discern what content is new or has been recently altered, *a ‘new’ glyph may be presented on folders and files that are new or have been recently changed.*” Shappell, ¶ 80; *id.* ¶ 81 (“This glyph provides a visual indicator of new content.”). A POSA would have understood this “new” glyph presented “on” “**recently** changed” files is and obviously would be presented **proximate the file icon**, as an additional overlay on the icon that may optionally extend beyond the underlying icon’s visible graphic, which may depend on the size/space of the underlying icon.²⁰ For example, Barber

²⁰ A POSA would have understood “proximate” an icon to encompass overlays of distinct graphics, including those that may overlap at least in part with the underlying icon. *E.g.*, EX1023, European Patent Application Publication No. EP 1,621,990 A2

(EX1024, U.S. Pat. No. 5,923,325) discloses a graphical availability indicator (boxed in red) associated with and proximate to a file icon (boxed in green) that does not overlap with the file icon:

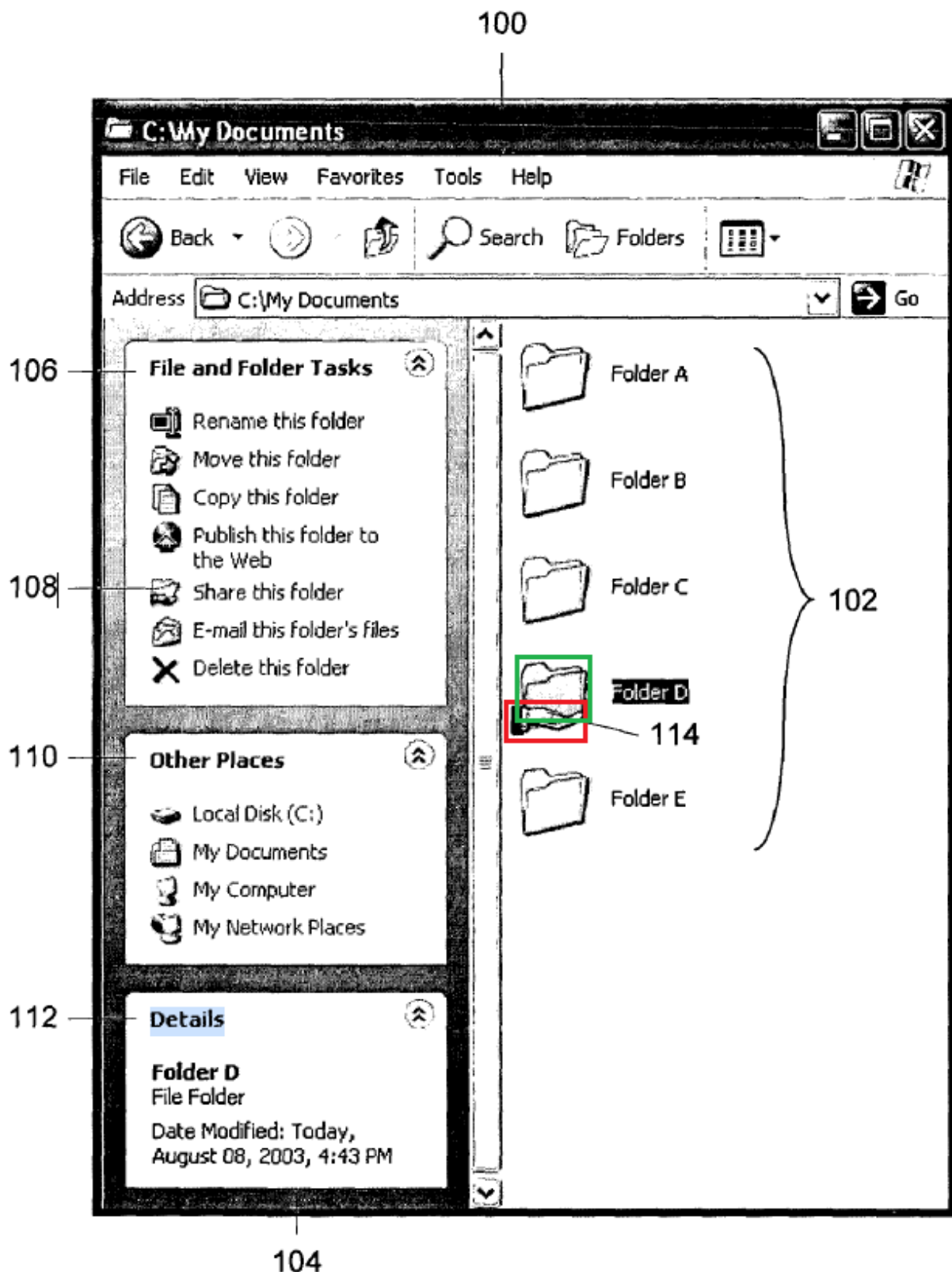


Barber, Fig. 5 (annotated); see also *id.*, 6:6-9 (“In FIG. 5, the object 52 is selected for printing. The first help image illustrates that a mouse 56 is manipulated so as to move a cursor, represented by an arrow 58, onto the object 52.”). This is one obvious variant of a graphical availability indication that is proximate (i.e., near) a file icon

to Kitamaru, et al., ¶ 0084 (describing that “the window 5113 is displayed **near** the icon 5105” where window 5113 partly overlaps with icon 5105).

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

but not overlapping the file icon. Another obvious variant would be to overlay a graphical availability indicator such that it slightly overlaps with an underlying file icon that is proximate to the graphical availability indicator, as shown in the following example from Figure 1 of Moromisato, showing a hand overlay graphical availability indicator (boxed in red) overlapping an associated file icon (boxed in green):



Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Moromisato, Fig 1 (annotated); *see also id.*, 3:15-17 (“When a selected folder is designated as shared, a ‘hand’ overlay is applied to the icon that represents the folder as shown at 114 in FIG. 1.”).

657. As another illustration, where a “missing” file is currently unavailable (for example, its hosting system is offline), “a display icon 1506 is presented as a ghosted icon that includes a small image of a red ‘X’ **or other suitable visual indicia** that the file is unavailable to the user.” Shappell, ¶ 79. The red “X” is presented proximate the file icon. Considered as a whole, Shappell’s teachings of the “new” glyph indicating files that “have been recently changed,” red “X” indicating file **unavailability**, and suggestions to consider “other suitable visual indicia” and “intuitive” GUI extensions, would have prompted a POSA to consider analogous icon-proximate indications for the **availability** of a recently modified “missing” file. The “ghosted” icon **1504** is semi-transparent to indicate a “missing” file that has been modified on another computer, but that “ghosted” icon itself would not inform the user whether the “missing” file was “**recently** changed” or not. *Id.*, ¶¶ 79-81. A POSA would have considered the “new” glyph a trivially obvious design-choice to present proximate the “ghosted” icon for “**recently** changed” “missing” files (for example, using a time threshold to limit the duration of display of such a graphic based on recency), leveraging the “modification date” in the metadata. *Id.*, ¶¶ 78,

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

80, 81. The glyph would intuitively inform the user, using a GUI feature already present in Shappell's system: a "new" glyph on the icon of "recently changed" downloaded files and a "new" glyph on the "ghosted" icon for recently modified "missing" files available for download, disclosing **a graphical availability indication proximate a file icon**. *Id.*

658. Shappell's exhortations to exploit and intuitively extend Windows GUI features would have motivated numerous other **icon-proximate indicators of file availability** obviously supplementing or replacing Shappell's "ghosted" icon. Numerous icon-proximate indicators were present in Windows for consideration. For example, Shappell describes that "the user may right-click an icon" to "present a graphical interface" such as "Context Menu 1700" shown in Figure 17. Shappell, ¶ 83, Fig. 17. In this familiar Windows feature, a user right-clicks a file icon, prompting display proximate the icon of a menu of options regarding the file. Shappell does not describe what would happen if a user right-clicked a "ghosted" icon indicating file availability, leaving it to the designer to populate the pop-up "Context Menu" menu options. A POSA would have found it obvious that the pop-up menu would **indicate file availability proximate the icon**, such as displaying a "Download" menu option caused by the receipt of the metadata. In fact, another prior art Microsoft file-synchronization patent (Moromisato) describes precisely

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

such a pop-up “Download” menu option, proximately indicating availability to
“Download” a file. Moromisato, 15:36-43 (a “menu with appropriate actions would
then appear,” including “Download”).

659. A POSA similarly would have been motivated to exploit “[t]he
‘Details’ view in Windows Explorer [that] displays columns of information about
each file in the file system” (Moromisato, 17:14-21), as another obvious preexisting,
convenient, and user-friendly Windows feature to graphically indicate, **proximate a
representation of the “missing” file**, the significant information of which files are
available for download—again, illustrated by Moromisato (*see id.*), complementing
Shappell’s own suggestion to use “Windows Explorer-style” displays (Shappell,
¶ 42).

660. A POSA also would have considered obvious, as described in
Moromisato, that the icon “that represents that file is ***provided with an icon overlay***”
that “can be similar to ***the ‘hand’ overlay 114 (FIG. 1) displayed by Windows XP
Pro*** file system when a folder is shared over the network.” Moromisato, 17:3-14,
15:14-22 (where synchronized “***file downloading is deferred to a later time***, that file
is represented in the OS file system as a ‘stub’ file that is very similar to a Windows
shortcut,” with “the same icon as the target file, but it ***may have an overlay of some
sort***”), Fig. 1. This familiar Windows “hand” overlay for a network-shared resource

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

discloses another obvious **graphical availability indication proximate a file icon** to be presented proximate a ghosted (or alternatively non-ghosted) icon for a “missing” file that is available for download. The classic Windows “shortcut” icon overlay (or a similar adjacent indicator) also would have been another trivially obvious choice to indicate, **proximate the file’s icon**, that the file is not present locally but is available for download. *Id.*, 15:14-22 (where synchronized “**file downloading is deferred to a later time**, that file is represented in the OS file system as a ‘stub’ file that is **very similar to a Windows shortcut**,” with “the same icon as the target file, but it **may have an overlay of some sort**”).

661. Yet another obviously well-suited, familiar Windows feature to supplement or replace Shappell’s “ghosted” icon was “Infotips”—a pop-up graphic “to display extended information” **proximate a file icon** when mousing over it, again naturally applying Shappell’s motivation to use existing Windows GUI features to provide information about remotely-located files. Moromisato, 17:22-29 (illustrating “infotips” in Windows for displaying information about “synchronized files”).

662. Another obvious “intuitive” extension, inspired by the **negative-connotation** red “X” overlay for **unavailability** of a “missing” file, would have been a **positive-connotation** overlay proximate the “ghosted” icon for **availability** for

download, such as a green checkmark (✓).

663. The motivation to combine Brown and Shappell above in [1c] applies with full force here. Brown contemplates standard client computers with Microsoft software but does not expressly describe GUI features. Brown, 7:3-11. Shappell picks up where Brown leaves off, motivating a POSA to consider Windows and Windows-extending GUI features to improve upon Brown with Shappell's teachings to graphically indicate to Brown system users the existence and status of modified files, with full expectation of predictable success.

h. [1g]: “wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device”

664. Brown in combination with Shappell discloses wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device.

665. For the reasons discussed above in connection with claim element [1e] and the other foregoing limitations of claim 1, Brown in combination with Hesselink discloses operating systems having graphical user interfaces and functionality for sending updated files from one client to a server and, in turn, from the server to

another client. In addition, for the reasons discussed above in connection with claim element [1e], it would have been obvious to look to references like Shappell for implementations of user interface indications that updated versions of files are available for download.

666. As discussed above for claims [1e] and [1f], at least one of Shappell's disclosed and obviously motivated indicator for an available "missing" file **"indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded."** Shappell's indicator of a "missing" modified file (including as obviously implemented proximate a file icon, as per claim [1f]) may represent that the file is **"available"** to be downloaded ***from another system*** to the computer displaying the indicator. Shappell, ¶¶ 79, 81, 62. In Brown's client-server system, that "other system" is server **105**. The same motivations to combine Brown with Shappell apply as discussed above.

667. In the modified Brown system incorporating Shappell's teachings, the graphical availability indication would indicate that the updated version of the file is both available for download at the server and present at the originating client device (for example, desktop computer **130**), after the originating client device uploaded a modified file to the server. This could occur, for example, when client

130 completes a full synchronization iteration by first transmitting its metadata to the server and then transmitting an updated file to the server before client **135** begins a synchronization iteration and subsequently downloads the metadata describing the file that was updated at client **130** before downloading the actual file itself.

i. [1h]: “subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device”

668. For the reasons above in connection with element [1c], and the reasons below, Brown in combination with Shappell and Rao discloses and renders obvious, subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device.

669. As discussed in connection with element [1c] above, Brown discloses that as part of the synchronization process, the server automatically transfers the file metadata to the synchronizing client device prior to transferring the copy of the file. For the reasons discussed for element [1c] above, Brown in view of Shappell and Rao further renders obvious that the copy of the modified first file is transferred subsequent to the metadata based on a priority assignment configuration at Brown’s server.

- j. [1i]: “wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration”

670. Brown in view of Shappell and Rao discloses and renders obvious a server system or first client device that comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

671. For the reasons above in connection with element [1c], Brown in view of Shappell and Rao discloses and renders obvious that metadata associated with a modified file is **assigned a greater priority** for transmission to/from the client from/to the server than a **lower priority** assigned to a copy of the file itself, based on a priority assignment configuration configured at the server that assigns a first, greater priority to metadata and a second, lower priority to files.

2. Claim 2

672. Claim 2 depends from claim 1 and further recites “**wherein, before the**

copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.” The limitations of claim 2 are disclosed by and obvious over the disclosures in Brown in combination with Hesselink and Shappell.

673. For the reasons above for claim 1 and the reasons below, Brown and Hesselink in combination with Rao and Shappell discloses and renders obvious claim 2. As discussed above for [1e]-[1f], Shappell renders obvious modifying Brown so that, **before the copy of the first file is transferred to the second client device, the transfer of the first metadata causes a file representation (icon) on the user interface of the second device to be updated based on the first metadata.** As discussed for [1e]-[1f], Shappell teaches that the receipt of metadata from another computer’s “modification of a file” “will *change the visual representation of the file* to a ‘missing’ file” available for download. (Shappell, ¶ 81.) All of the obvious GUI design options for presenting the indicator proximate the icon discussed for [1e]-[1f] apply equally here, all of which update the file representation (icon).

3. Claim 3

674. Claim 3 depends from claim 1 and further recites “**wherein the**

computer program instructions, when executed, cause the server system to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.” The limitations of claim 3 are disclosed by and obvious over the disclosures in Brown in combination with Hesselink, Shappell, and Rao.

- b. [3a]: “receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;”**

675. As discussed above in connection with elements [1b] and [1c], Brown’s

client devices (such as clients **130** and **135** in Figure 1) are fungible in the sense that they must all communicate with the server in the same way (first transmitting updated metadata back and forth between the server to determine which files have changed and then, as necessary, obtaining copies of the modified files themselves). Thus, for the same reasons above in connection with [1b] and [1c], this limitation would have been obvious.

c. [3b]: “determine that the server system is in communication with the first client device associated with the user; and”

676. Brown in view of Hesselink discloses or renders obvious determining that the server system is in communication with the first client device associated with the user.

677. For example, it would have been obvious to synchronize the server with a client device, for example the laptop computer described in Brown. The client would initiate communication with the server, and the server could respond based on a determination that the client and server are in communication. Also, it would have been obvious to use any handshake type function in TCP/IP or HTTP or any protocol, given that Brown teaches that TCP/IP and HTTP are used, or any protocol may be used: “The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. ***Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.***” Brown, 3:1-8.

678. This element is also obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission to the offline client. Hesselink, ¶ 183 (“As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. Computers that are offline at such a time will be automatically updated as soon as those computers go online again.”); *id.*, ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”). A POSA would have been motivated to improve Brown’s system, to

the extent it was deficient, with this improvement from Hesselink for the reasons discussed above in connection with claim 1 and, in particular, in view of Brown's stated goal to provide "a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made." Brown, 1:46-50, 3:42-46.

- d. **[3c]: "automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device."**

679. As discussed above in connection with elements [1b] and [1c], Brown's clients are fungible in the sense that they must all communicate with the server in the same way (first transmitting updated metadata back and forth between the server to determine which files have changed and then, as necessary, obtaining copies of the modified files themselves).

680. This element is disclosed and/or rendered obvious in view of Brown in combination with Hesselink. For example, it would have been obvious to synchronize the server with a client device, for example the laptop computer described in Brown. The client would initiate communication with the server, and the server could respond based on a determination that the client and server are in

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

communication. Also, it would have been obvious to use any handshake type function in TCP/IP or HTTP or any protocol, given that Brown teaches that TCP/IP and HTTP are used, or any protocol may be used: “The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. ***Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.***” Brown, 3:1-8.

681. Brown teaches that the synchronization is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server. ***Client machines synchronize with each other by synchronizing to a common server account.***” *Id.*, 2:63-67.

682. A POSA would have understood and found it obvious that a second file on the second client device (for example, laptop computer **135**) would be received from the server when that file is modified. Brown describes a synchronization process with respect to desktop client **130** that starts by checking whether the server has any updates for the user’s account, which would be updates from other client

devices of that user such as laptop client **135**. Brown, 10:18-27 (“Otherwise, server 105 has changes that client 130 lacks. Server 105 then sends the SSD to client 130 (in response to a request for the SSD by the client), informing the client of the pertinent changes, as shown in box 415. Specifically, the SSD includes the SSI, the SIDs of any directories that contain changes since the last time client 130 synchronized with server 105, the SIDs of all items (files and directories) in the changed directories, and the metadata of all items (files and directories) that have been changed since the last time client 130 synchronized with server 105.”). Later, client **130** can send all pertinent changes made on client **130** to server **105** to update other client devices associated with the same user, as would be motivated by a user wanting to keep, for example, work and home computers synchronized as discussed in Brown. *Id.*, 10:53-55 (“Once client 130 has downloaded all the pertinent changes from server 105, client 130 can send all the pertinent changes made on client 130 to server 105.”).

683. Brown teaches that the server application (SFS) automatically transfers the modified file received from the second device to the third device to replace an older version of the file. As discussed above with respect to element [1c], it would have been obvious to update another device associated with the same user by transferring an updated version of a file received from one user device to the other.

Synchronization in Brown's system is bidirectional between multiple client devices that "synchronize with each other by synchronizing to a common server account."

Brown, 2:65-67. "Client machines synchronize with each other by synchronizing to a common server account." *Id.*, 2:63-67. After the server account receives a modified file from one client device, the server application transfers that file to another client device associated with the common server account. *Id.*, 10:18-27, 10:53-55 (describing that when the "server 105 has changes that client 130 lacks," then the client will "download[] all the pertinent changes from server 105").

684. Brown expressly teaches that a downloaded file **replaces an older version of the file** if it exists: "On the other hand, if less than a threshold number of message digests in the MDAs match, then *the entire file is downloaded from the server*. Once the download file is completely constructed, the client inserts the download file into its final location, *replacing an older version of the file if it exists*." Brown, 13:18-23.

685. This element is also obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission to the offline client. Hesselink, ¶ 183 ("As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the

update has been performed at one location. Computers that are offline at such a time will be automatically updated as soon as those computers go online again.”); *id.*, ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”).

686. In addition, a POSA would have found it obvious in view of Hesselink to use Brown’s server to automatically push updates from the server to connected clients as soon as file updates occur. For example, Hesselink discloses: “As noted above, ***when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.*** Computers that are offline at such a time will be automatically updated as soon as those computers go online again.” Hesselink, ¶ 183. A POSA would have been motivated to improve Brown’s system, to the extent it was deficient, with this

improvement from Hesselink for the reasons discussed above in connection with claim 1 and, in particular, in view of Brown's stated goal to provide "a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made" and Brown's teaching that "a person skilled in the art [would] recognize that *the client SA can use any scheduling algorithm to initiate synchronization*," including Hesselink's algorithm for automatic updating. Brown, 1:46-50, 3:42-46.

4. Claim 4

687. Claim 4 depends from claim 1 and further recites "**wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.**" The limitations of claim 4 are disclosed by and obvious over the disclosures in Brown in combination with Hesselink and Shappell.

688. The embodiment in Figure 1 discloses server **105**, which includes a Synchronization File System (SFS), labeled SFS **110**. "The server component is a server Synchronization File System (SFS) with a functional interface and metadata organization that enables efficient and reliable synchronization by the clients." Brown, 2:59-62. "Typically, server SFS **110** is installed on a hard disk drive within

server **105**” *Id.*, 3:59-60.

689. Figure 1 from Brown also discloses client devices, such as desktop computer **130** and portable computer **135**.

690. The server **105** is in communication with the two client devices that each have a synchronization application (SA) installed on them, as illustrated in Figure 1. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server.” Brown, 2:63-65. “Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.” *Id.*, 3:6-8.

691. For the reasons above in connection with element [1c], Brown discloses and renders obvious that the two client devices and server are each associated with the same user. Brown contemplates “multiple clients of a single user account” that may “push changes to the server concurrently” and that “the server SFS interleaves the changes.” Brown, 4:62-64.

692. Brown discloses that its clients submit push requests to the server described in Brown. Brown, 15:30-35 (“FIGS. 11A–11F show a flowchart of the procedure used to **push** changes to the server from a client of FIG. 1, according to an embodiment of the invention. At step 1105, the client gets the first change to **push** to the server. At step 1107, the client checks to see if the change is a file to

upload to the server.”); *Id.*, Abstract (“Distributed clients access the server, to learn about changes made to the files on the server, and to push local changes of the files onto the server.”). For the reasons discussed in connection with element [1b], pushing changes as they occur would have been obvious in view of Brown in combination with Hesselink and Shappell.

5. Independent Claim 8

693. Claim 8, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 8 (method)
1. A system, comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	8. A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated	receiving, over a network, from the first client device, first metadata associated

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and
(ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and	(ii) the graphical availability indication is presented proximate a graphical file representation of the first file on a user interface of the second client device, and
wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and	wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and
subsequent to the transfer of the first metadata to the second client device,	subsequent to the transfer of the first metadata to the second client device,

transfer the copy of the first file to the second client device,	transferring the copy of the first file to the second client device,
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

694. The only substantive difference between claim 1 and claim 8 is that the latter recites: “A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method.” For the reasons discussed above in connection with claim 1, claim 8 would have been obvious.

6. Claim 9

695. Claim 9 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 2:

Claim 2 (system)	Claim 9 (method)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client	9. The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client

device to be updated based on the first metadata.	device to be updated based on the first metadata.

696. Claim 9 presents no material difference from claim 2. Claim 9 is thus obvious for the same reasons as claim 2, as discussed above.

7. Claim 10

697. Claim 10 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 10 (method)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	10. The method of claim 8, further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client

device and (ii) receiving the copy of the second file from the second client device.	device and (ii) receiving the copy of the second file from the second client device.

698. Claim 10 presents no material difference from claim 3. Claim 10 is thus obvious for the same reasons as claim 3, as discussed above.

8. Claim 11

699. Claim 11 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 11 (method)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	11. The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

700. Claim 11 presents no material difference from claim 4. Claim 11 is thus obvious for the same reasons as claim 4, as discussed above.

9. Independent Claim 13

701. Claim 13, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system,**” cause operations, instead of a “**system.**” As the following table shows, both claims

otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 13 (media)
1. A system, comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	13. One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client	wherein, before the copy of the first file is transferred to the second client

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

device:	device:
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	(i) the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and
(ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and	(ii) the availability indication is presented proximate a graphical file representation of the first file on a user interface of the second client device, and
wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and	wherein the availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and
subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,	subsequent to the transfer of the first metadata to the second client device, transferring the copy of the first file to the second client device,
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

702. The only difference between claim 1 and claim 13 is that the latter

recites “One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising” the same steps of claim 1. For the reasons discussed above in connection with claim 1, and reasons below, claim 13 would have been obvious.

703. Brown discloses that its server SFS application is “[t]ypically” “installed on a hard disk drive within server 105” but notes that “a person skilled in the art will recognize other forms of media . . . or optical media” would be appropriate. Brown, 3:59-62. A POSA would have understood that a hard disk drive includes “[o]ne or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations” on Brown’s server.

10. Claim 14

704. Claim 14 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 2:

Claim 2 (system)	Claim 14 (media)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.	14. The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the first file presented on the user interface of the second client device to be updated based on the first metadata.

705. Claim 14 presents no material difference from claim 2. Claim 14 is thus obvious for the same reasons as claim 2, as discussed above.

11. Claim 15

706. Claim 15 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 15 (media)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	15. The media of claim 13, the operations further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the

second file from the second client device.	second file from the second client device.

707. Claim 15 presents no material difference from claim 3. Claim 15 is thus obvious for the same reasons as claim 3, as discussed above.

12. Claim 16

708. Claim 16 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 16 (media)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	16. The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

709. Claim 16 presents no material difference from claim 4. Claim 16 is thus obvious for the same reasons as claim 4, as discussed above.

C. '787 Ground 3: Claims 1-4, 8-11, and 13-16 of the '787 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink, Shappell, and Rao.

1. Independent Claim 1

710. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with

computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and

[1d] automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,

[1e] wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and

[1f] (ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and

[1g] wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

on the priority assignment configuration.

711. As shown in the table below, claim 1 of the '787 patent is similar to claim 1 of the '823 patent (substantively identical limitations appear on the same row).

'823 Claim 1	'787 Claim 1
1. A system, comprising:	1. A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the	automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	
	(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
(ii) the graphical availability indication is presented proximate a file icon representing the first file on a user interface of the second client device, and	
	(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and
wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded	

from the server system to the second client device; and	
subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,	
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

712. The limitations that are substantively identical between claim 1 of the '823 patent and claim 1 of the '787 patent would have been obvious for the reasons discussed above in Ground 5 for the '823 patent.

- b. **[1e]: “wherein, before the copy of the first file is transferred to the second client device: (i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and”**

713. Brown and Hesselink in combination with Shappell discloses that, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a file representation of the

first file presented on a user interface of the second client device to be updated based on the first metadata.

714. For the reasons discussed above in connection with '823 claims [1e] and [1f], it would have been obvious to transfer the first metadata before the copy of the first file is transferred to the second client device, and it would have been obvious to look to reference like Shappell in view of Brown's brief reference to Windows operating systems that implemented user interfaces to find more detail about potential user interface features that could be used to indicate to users the state of files before, during, and after synchronization.

715. In addition, it would have been obvious in view of Brown in combination with Hesselink and Shappell to **transfer the first metadata to the second client device to cause a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata.**

716. Shappell discloses ways to "***inform a user of a group shared space file that is not present locally***" such that "***display icons are presented to the user.***"

Shappell, ¶ 79. Shappell describes:

FIG. 15 illustrates a graphical interface 1500 for a Shared Space folder that contains files that are not local. As shown, ***a left hand pane 1502 presents a display icon 1504 to denote a 'missing' file that is available on another system. This icon is ghosted or semi-transparent.*** As is

shown in the flow chart in FIG. 20, ***the system renders such a display upon receipt that metadata concerning a shared file is available in the peer network***, as is shown in method step 2002. Other files that are ‘missing’ may be unavailable to the user, such as when ***a system that is hosting the file of interest is offline***. In this instance, a display icon 1506 is presented as a ghosted icon that includes ***a small image of a red “X” or other suitable visual indicia*** that the file is unavailable to the user.

Id. This excerpt describes at least two examples of **updates to file representations**.

First, an “icon” representing a missing file may be “ghosted or semi-transparent” to indicate that an updated file is “available on another system.” *Id.* Second, the “small image of a red ‘X’” on a file’s icon indicates that the file is available on another “system that is hosting the file of interest” but is currently “offline.” *Id.*

717. Shappell further discloses that these file representations are updated “**upon receipt**” of “**metadata concerning a shared file**” that “**is available.**” Shappell, ¶ 79; *see also id.*, ¶ 81 (“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will remove the local copy of the out-of-date file. ***This will change the visual representation of the file to a “missing” file within the shared space folder*** when the application setting does not automatically update files, as is shown in step 2010 in FIG. 20. On the other hand, if automatic replication of the shared file is enabled,

the local machine obtains an updated version of the file, as is shown in step 2006 in FIG. 20. Finally, an appropriate visual indicator is presented to the user as shown in step 2008.”); *see also id.* ¶ 76 (“For sharing files among group members, ***metadata is transmitted to all members when files become available***, such as is shown in the steps 1904 and 1906 in the flow chart in FIG. 19. ***Such metadata are sufficient to render an icon of the shared file*** and may include date, time, thumb-nail information, name size, and optionally the source of the information.”). For the reasons above and in connection with ’823 claims [1e] and [1f], it would have been obvious in view of these disclosures for **the transfer of the first metadata to the second client device to cause a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata.**

718. The same motivations to combine Brown and Hesselink with Shappell and Rao described above for ’823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Shappell into Brown’s systems and methods to practice claim [1e].

- c. **[1f]: “(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and”**

719. Brown and Hesselink in combination with Shappell disclose that instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device.

720. For the reasons discussed above in connection with claim [1e], Brown discloses or suggests a graphical user interface, and it would have been obvious to look to references like Shappell for implementations of user interface file representations related to file synchronization functionality.

721. As discussed above for claim [1e], Shappell describes at least two examples of **updates to file representations**. Shappell, ¶ 79. First, an “icon” representing a missing file may be “ghosted or semi-transparent” to indicate that an updated file is “available on another system.” *Id.* Second, the “small image of a red ‘X’” on a file’s icon indicates that the file is available on another “system that is hosting the file of interest” but is currently “offline.” *Id.* Both of these examples

describe that, **instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device.** In particular, both exemplary updated file representations are expressly described in Shappell as representing versions of files “available on another system” or available on another “system that is hosting the file,” respectively. *Id.* Shappell further discloses that, upon learning of an updated version of a file, Shappell’s clients may “remove the local copy of the out-of-date file.” *Id.* ¶ 81. Thus, the updated file representation would represent a version of the file that is stored on another client (e.g., the originating “system that is hosting the file of interest”), and it would have been obvious for the file representation to represent the updated version of the first file that is currently stored on the first (originating) client device and not currently stored on the second (receiving) client device.

722. For example, as discussed above in connection with ’823 Ground 1 (e.g., for claims [1c] and 3), it would have been obvious to attempt to keep files stored on Brown’s **first and second client devices** synchronized and up-to-date by using Brown’s server to push changes from one client to the other. Thus, it would

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

have been obvious to indicate on a user interface on Brown's **second client device** that the most recent version of a file is currently stored on Brown's **first client device** (the device where the file was modified) upon receipt of **first metadata**.

723. For the reasons above and in connection with '823 claim [1e] and [1f], it would have been obvious to incorporate the abovementioned functionality from Shappell into the systems and methods of Brown to include a change of visual representation of a file to indicate that the file has recently been modified when a metadata-based notification of a modification of a file contained in a shared space occurs to indicate the file is available on another client—for example, the server in Brown could send metadata to the **second client device** to indicate that an updated version of a file is available on the **first client device** prior to actually sending the file, and Shappell's user interface functionality could be used to properly notify the user of this situation, as discussed above.

724. The same motivations to combine Brown and Hesselink with Shappell and Rao described above for '823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Shappell into Brown's systems and methods to practice claim [1f].

2. Claim 2

725. Claim 2 depends from claim 1 and further recites “**wherein, before the**

copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.” Aside from the reference to the “second client device” having a “user interface,” this limitation is substantively identical to ’823 element [1e] and would have been obvious in view of Brown, Hesselink, and Shappell for the reasons discussed above in connection with ’823 element [1e]. As discussed above for the ’823 patent, Brown discloses that its clients may have user interfaces—for example, user interfaces implemented as part of Microsoft operating systems, and it would have been obvious to combine Shappell with Brown to implement this claim. Brown, 7:3-11.

3. Claim 3

726. Claim 3 depends from claim 1 and further recites “**wherein the computer program instructions, when executed, cause the server system to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the**

content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.” The limitations of ’787 claim 3 are substantively identical to those of ’823 claim 3 and would have been obvious for the reasons discussed above in connection with ’823 claim 3.

4. Claim 4

727. Claim 4 depends from claim 1 and further recites “**wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.**” The limitations of ’787 claim 4 are substantively identical to those of ’823 claim 4 and would have been obvious for the reasons discussed above in connection with ’823 claim 3.

5. Independent Claim 8

728. Claim 8, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**.” As the following table

shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 8 (method)
A system comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and	receiving, over a network, from the first client device, first metadata associated with the updated version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file	wherein, before the copy of the first file

is transferred to the second client device:	is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and	(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and	(ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

729. The only substantive difference between claim 1 and claim 8 is that the latter recites “[a] method” for performing the same steps recited in claim 1. Thus, for the reasons discussed above in connection with claim 1, claim 8 would have been obvious.

6. Claim 9

730. Claim 9 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 2:

Claim 2 (system)	Claim 9 (method)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.	9. The method of claim 8, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

731. Claim 9 presents no material difference from claim 2. Claim 9 is thus obvious for the same reasons as claim 2, as discussed above.

7. Claim 10

732. Claim 10 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 10 (method)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	10. The method of claim 8, further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second

file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

733. Claim 10 presents no material difference from claim 3. Claim 10 is thus obvious for the same reasons as claim 3, as discussed above.

8. Claim 11

734. Claim 11 depends from claim 8 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 11 (method)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	11. The method of claim 8, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

735. Claim 11 presents no material difference from claim 4. Claim 11 is thus obvious for the same reasons as claim 4, as discussed above.

9. Independent Claim 13

736. Claim 13, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system,**” cause operations, instead of a “**system.**” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 13 (media)
A system comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receiving, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file	receiving, over a network, from the first client device, first metadata associated with the updated version of the first file

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and	that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; and
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transferring, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,
wherein, before the copy of the first file is transferred to the second client device:	wherein, before the copy of the first file is transferred to the second client device:
(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and	(i) the transfer of the first metadata to the second client device causes a file representation of the first file presented on a user interface of the second client device to be updated based on the first metadata, and
(ii) instead of the updated file representation of the first file representing a version of the first file currently stored on the second client device, the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and	(ii) the updated file representation represents the updated version of the first file that is currently stored on the first client device and not currently stored on the second client device, and
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least

one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

737. As shown, the only difference between claim 1 and claim 13 is that the latter recites “One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising” the same steps of claim 1. For the reasons discussed above in connection with claim 1, and reasons below, claim 13 would have been obvious.

738. Brown’s server includes one or more processors programmed with computer program instructions that may be executed to cause operations. Brown, 3:57-59 (“Server 105 includes all the typical elements of a server, such as a central **processor**, memory, bus, disk space, etc.”). As discussed above in connection with ’823 claim [1a], Brown and Hesselink both disclose servers containing hard disk drives, which could be used to implement this claim. In particular, Brown discloses that its server SFS application is “[t]ypically” “installed on a hard disk drive within server 105” but notes that “a person skilled in the art will recognize other forms of media . . . or optical media” would be appropriate. Brown, 3:59-62. A POSA would have understood that a hard disk drive includes “[o]ne or more non-transitory machine-readable media storing instructions that, when executed by one or more

processors of a server system, cause operations” on Brown’s server.

10. Claim 14

739. Claim 14 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 2:

Claim 2 (system)	Claim 14 (media)
2. The system of claim 1, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.	14. The media of claim 13, wherein, before the copy of the first file is transferred to the second client device, the transfer of the first metadata to the second client device causes an availability indication of the updated version of the first file to be presented on the user interface of the second client device based on the first metadata.

740. Claim 14 presents no material difference from claim 2. Claim 14 is thus obvious for the same reasons as claim 2, as discussed above.

11. Claim 15

741. Claim 15 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 3:

Claim 3 (system)	Claim 15 (media)
3. The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	15. The media of claim 13, the operations further comprising:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the	receiving a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the

second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determining that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transferring the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

742. Claim 15 presents no material difference from claim 3. Claim 15 is thus obvious for the same reasons as claim 3, as discussed above.

12. Claim 16

743. Claim 16 depends from claim 13 and, for purposes of my analysis, is not materially different from claim 4:

Claim 4 (system)	Claim 16 (media)
4. The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user	16. The media of claim 13, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user

modifying the content of the first file stored on the first client device.	modifying the content of the first file stored on the first client device.

744. Claim 16 presents no material difference from claim 4. Claim 16 is thus obvious for the same reasons as claim 4, as discussed above.

D. '607 Ground 3: Claims 1, 3-5, 7, 12, 14, 17, 19, and 20 of the '607 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink, Shappell, and Rao.

1. Independent Claim 1

745. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

1[c] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1d] determine that the server system is not in communication with a second client device associated with the user;

[1e] store the copy of the first file on the server system;

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

[1f] automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[1g] automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

746. As shown in the table below, claim 1 of the '607 patent is similar to claim 1 of the '823 patent (substantively identical limitations appear on the same row).

'823 Claim 1	'607 Claim 1
1. A system comprising:	1. A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being an updated version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, over a network, from the first client device, first metadata associated with the updated version of the first file	receive, from the first client device, first metadata associated with the version of the first file that is generated from the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
	determine that the server system is not in communication with a second client device associated with the user;
	store the copy of the first file on the server system;
automatically transfer, based on the first priority being greater than the second priority, the first metadata over a network to a second client device associated with the user such that the first metadata is transferred to the second client device before the copy of the first file is transferred to the second client device,	automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
	automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.
wherein, before the copy of the first file is transferred to the second client device:	
(i) the transfer of the first metadata to the second client device causes a graphical availability indication of the updated version of the first file to be presented at the second client device based on the first metadata, and	
(ii) the graphical availability indication	

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

is presented proximate a file icon representing the first file on a user interface of the second client device, and	
wherein the graphical availability indication indicates that the updated version of the first file generated from the user modifying the content of the first file is available to be downloaded from the server system to the second client device; and	
subsequent to the transfer of the first metadata to the second client device, transfer the copy of the first file to the second client device,	
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	

747. The limitations that are substantively identical between claim 1 of the '823 patent and claim 1 of the '607 patent would have been obvious for the reasons discussed above in Ground 5 for the '823 patent.

b. [1d]: “determine that the server system is not in communication with a second client device associated with the user;”

748. This limitation would have been obvious in view of Brown in combination with Hesselink. Hesselink discloses repeatedly checking whether other devices are in communication to synchronize. This type of determination would have been obvious to use with Brown, to account for the situation in Brown where the second client device is not currently online. In the Brown system as obviously modified by Hesselink such that the modified file would be automatically transferred responsive to the modification from the first client through the server to the second client, when the second client device is not currently online, it would have been obvious to determine whether or not the server is in communication with the second client, to determine whether or not the file could be transferred to the second client.

749. For example, Hesselink discloses: “As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.*** The persistence of the system ensures that updates are completed for all online computers, as if there is some failure or problem during the update, the

connection restores itself to ensure that the update is completed.” Hesselink, ¶ 183.

“With this functionality, when the user works on a file, it is automatically updated in the remote device by the device module. Once the file has been put into the cache memory on the local computer, the user can work locally on the file via the user module, as described above. Therefore if the connection to the network is lost, this will not affect the user’s ability to work on the file. Updates to the official reference file (and optionally the file as located in other remote computers) obviously cannot be performed as long as the connection is broken. Furthermore, ***the system is persistent and may be configured on a user computer to automatically and repeatedly attempt to reconnect with any and all remote devices which are actively subscribed to when a connection is broken during an active subscription.*** The present software recognizes that the file is in cache and the user can continue working locally when the connection is broken. When the network connection is restored, the software automatically updates the home file (official reference file) on the remote device via the local user module’s transmission to the remote device module, as described above (and optionally, on any other computers where the file may reside). This feature is useful not only for when a network connection is accidentally broken, but in situations where the user moves to a location where the user knows a network connection is not possible, as in the example described above,

while flying on an airplane, or in some other location that is not network accessible. Even in these situations, as soon as the user logs into the network again, the official reference file may be automatically updated, so that when the user returns to the location of the home file (official reference file), the home file is already up to date.” Hesselink, ¶ 190.

750. The same motivations to combine Brown and Hesselink described above for ’823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Hesselink into Brown’s systems and methods.

c. [1e]: “store the copy of the first file on the server system;”

751. Brown discloses that the file uploaded from the client is stored at the server. In the Brown system as obviously modified in view of Hesselink, the same storage at the server would be maintained regardless of whether the file was then immediately transferred to the second client, or whether the second client was not currently online. “Once the file has been completely uploaded, the server then stores the file in the specified directory and updates the file metadata.” Brown, 13:1-3.

- d. **[1g]: “automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.”**

752. This element is disclosed and/or rendered obvious in view of Brown in combination with Hesselink. For example, it would have been obvious to synchronize the server with a client device, for example the laptop computer described in Brown. The client would initiate communication with the server, and the server could respond based on a determination that the client and server are in communication. Also, it would have been obvious to use any handshake type function in TCP/IP or HTTP or any protocol, given that Brown teaches that TCP/IP and HTTP are used, or any protocol may be used: “The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. ***Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.***” Brown, 3:1-8.

753. Brown teaches that the synchronization is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

common server account.” Brown, 2:65-67. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server. ***Client machines synchronize with each other by synchronizing to a common server account.***” *Id.*, 2:63-67.

754. A POSA would have understood and found it obvious that the first file would be transferred from the server to the second client device (for example, laptop client **135**) when that file is modified. Brown describes a synchronization process with respect to laptop client **135** that starts by checking whether the server has any updates for the user’s account, which would be updates from other client devices of that user such as desktop client **130**. Brown, 10:18-27 (“Otherwise, server 105 has changes that client 130 lacks. Server 105 then sends the SSD to client 130 (in response to a request for the SSD by the client), informing the client of the pertinent changes, as shown in box 415. Specifically, the SSD includes the SSI, the SIDs of any directories that contain changes since the last time client 130 synchronized with server 105, the SIDs of all items (files and directories) in the changed directories, and the metadata of all items (files and directories) that have been changed since the last time client 130 synchronized with server 105.”).

755. Brown teaches that the server application (SFS) automatically transfers the modified file from the server to the second client device to replace an older

version of the file. As discussed above with respect to claim [1c], it would have been obvious to update another device associated with the same user by transferring an updated version of a file received from one user device to the other. Synchronization in Brown's system is bidirectional between multiple client devices that "synchronize with each other by synchronizing to a common server account." Brown, 2:65-67. "Client machines synchronize with each other by synchronizing to a common server account." *Id.*, 2:63-67. After the server account receives a modified file from one client device, the server application transfers that file to another client device associated with the common server account. *Id.*, 10:18-27, 10:53-55 (describing that when the "server 105 has changes that client 130 lacks," then the client will "download[] all the pertinent changes from server 105").

756. Brown expressly teaches that a downloaded file **replaces an older version of the file** if it exists: "On the other hand, if less than a threshold number of message digests in the MDAs match, then *the entire file is downloaded from the server*. Once the download file is completely constructed, the client inserts the download file into its final location, *replacing an older version of the file if it exists*." Brown, 13:18-23.

757. This element is also obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission

to the offline client. Hesselink, ¶ 183 (“As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. Computers that are offline at such a time will be automatically updated as soon as those computers go online again.”); ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”).

758. In addition, a POSA would have found it obvious in view of Hesselink to use Brown’s server to automatically push updates from the server to connected clients as soon as file updates occur. For example, Hesselink discloses: “As noted above, ***when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.*** Computers that are offline at such a time will be automatically updated as

soon as those computers go online again.” Hesselink, ¶ 183. A POSA would have been motivated to improve Brown’s system, to the extent it was deficient, with this improvement from Hesselink for the reasons discussed above in connection with claim 1 and, in particular, in view of Brown’s stated goal to provide “a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made” and Brown’s teaching that “a person skilled in the art [would] recognize that ***the client SA can use any scheduling algorithm to initiate synchronization***,” including Hesselink’s algorithm for automatic updating. Brown, 1:46-50, 3:42-46.

759. This limitation would have been obvious in view of Brown in combination with Hesselink. Hesselink discloses repeatedly checking whether other devices are in communication to synchronize. This type of determination would have been obvious to use with Brown, to account for the situation in Brown where the second client device is not currently online. In the Brown system as obviously modified by Hesselink such that the modified file would be automatically transferred responsive to the modification from the first client through the server to the second client, when the second client device is not currently online, it would have been obvious to determine whether or not the server is in communication with the second client, to determine whether or not the file could be transferred to the second client.

760. For example, Hesselink discloses: “As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.*** The persistence of the system ensures that updates are completed for all online computers, as if there is some failure or problem during the update, the connection restores itself to ensure that the update is completed.” Hesselink, ¶ 183. “With this functionality, when the user works on a file, it is automatically updated in the remote device by the device module. Once the file has been put into the cache memory on the local computer, the user can work locally on the file via the user module, as described above. Therefore if the connection to the network is lost, this will not affect the user’s ability to work on the file. Updates to the official reference file (and optionally the file as located in other remote computers) obviously cannot be performed as long as the connection is broken. Furthermore, ***the system is persistent and may be configured on a user computer to automatically and repeatedly attempt to reconnect with any and all remote devices which are actively subscribed to when a connection is broken during an active subscription.*** The present software recognizes that the file is in cache and the user can continue

working locally when the connection is broken. When the network connection is restored, the software automatically updates the home file (official reference file) on the remote device via the local user module's transmission to the remote device module, as described above (and optionally, on any other computers where the file may reside). This feature is useful not only for when a network connection is accidentally broken, but in situations where the user moves to a location where the user knows a network connection is not possible, as in the example described above, while flying on an airplane, or in some other location that is not network accessible. Even in these situations, as soon as the user logs into the network again, the official reference file may be automatically updated, so that when the user returns to the location of the home file (official reference file), the home file is already up to date.” Hesselink, ¶ 190.

761. This element is further obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission to the offline client. Hesselink, ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the

network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”).

2. Claim 3

762. Claim 3 depends from claim 1 and further recites “**wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.**” Claim 3 is not materially different from ’823 element [1i], as shown in the table below.

’823 Claim 1(j)	’607 Claim 3
wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

763. Because claim 3 presents no material difference from '823 element [1i], claim 3 would have been obvious for the same reasons discussed above in connection with the '823 patent.

3. Claim 4

764. Claim 4 depends from claim 1 and further recites “**wherein availability of the version of the first file is presented at the second client device based on the first metadata.**” This limitation would have been obvious for the reasons discussed above in connection with '823 Ground 1, claim [1e].

4. Claim 5

765. Claim 5 depends from claim 1 and further recites “**wherein the server system is caused to: receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being a version of the second file that is generated from the user modifying the content of the second file; determine that the server system is in communication with the first client device associated with the user; and automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on**

the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.” Claim 5 is not materially different from ’823 claim 3, as shown in the table below.

’823 Claim 3	’607 Claim 5
The system of claim 1, wherein the computer program instructions, when executed, cause the server system to:	The system of claim 1, wherein the server system is caused to:
receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being an updated version of the second file that is generated from the user modifying the content of the second file;	receive a copy of a second file from the second client device associated with the user, wherein the copy of the second file is automatically received from the second client device responsive to the user modifying a content of the second file stored on the second client device, the copy of the second file being a version of the second file that is generated from the user modifying the content of the second file;
determine that the server system is in communication with the first client device associated with the user; and	determine that the server system is in communication with the first client device associated with the user; and
automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.	automatically transfer the copy of the second file to the first client device associated with the user to replace an older version of the second file stored on the first client device, responsive to (i) determining that the server system is in communication with the first client device and (ii) receiving the copy of the second file from the second client device.

766. Because claim 5 presents no material difference from '823 claim 3, claim 5 would have been obvious for the same reasons discussed above in connection with the '823 patent.

5. Claim 7

767. Claim 7 depends from claim 1 and further recites “**wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.**” Claim 7 is not materially different from '823 claim 4, as shown in the table below.

'823 Claim 4	'607 Claim 7
The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.	The system of claim 1, wherein the copy of the first file is automatically received from the first client device responsive to (i) a push request of the first client device and (ii) the user modifying the content of the first file stored on the first client device.

768. Because claim 7 presents no material difference from '823 claim 4, claim 7 would have been obvious for the same reasons discussed above in connection with the '823 patent.

6. Independent Claim 12

769. Claim 12, for purposes of my analysis, is substantially similar to claim

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

1 except that it recites a “**method**” instead of a “**system.**” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 12 (method)
A system comprising: a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	A method being implemented by a server system comprising one or more processors executing computer program instructions that, when executed, perform the method, the method comprising:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receiving, by the server system, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
determine that the server system is not in communication with a second client device associated with the user;	determining, by the server system, that the server system is not in communication with a second client device associated with the user;
store the copy of the first file on the server system;	store the copy of the first file on the server system;
automatically transfer the first metadata to the second client device based on the	automatically transferring, by the server system, the first metadata to the second

first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and	client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.	automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

770. The only substantive difference between claim 1 and claim 12 is that the latter recites “[a] method” for performing the same steps recited in claim 1 that are performed by the server system described in claim 1. Thus, for the reasons discussed above in connection with claim 1, claim 12 would have been obvious.

7. Claim 14

771. Claim 14 depends from claim 12 and further recites “**wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the**

second priority to the copy of the first file based on the priority assignment configuration.” Claim 14 is not materially different from claim 3, as shown in the table below.

Claim 3 (system)	Claim 14 (method)
The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	The method of claim 12, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

772. Because claim 14 presents no material difference from claim 3, claim 14 would have been obvious for the same reasons discussed above in connection with claim 3.

8. Independent Claim 17

773. For reference, claim 17 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device

associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[1c] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration;

[1d] determine that the server system is not in communication with a second client device associated with the user;

[1e] store the copy of the first file on the server system;

[1f] automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[1g] automatically transfer, over a network, the copy of the first file to the second client device associated with the user, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device,

[1h] wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

774. Claim 17 is similar to claim 1, as shown in the table below
 (substantively identical limitations appear on the same row):

Claim 1 (system)	Claim 17 (system)
A system comprising:	A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file,
	wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

	priority to the copy of the first file based on the priority assignment configuration;
determine that the server system is not in communication with a second client device associated with the user;	determine that the server system is not in communication with a second client device associated with the user;
store the copy of the first file on the server system;	store the copy of the first file on the server system;
automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and	automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.	automatically transfer, over a network, the copy of the first file to the second client device associated with the user, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device,
	wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.

- b. **[17c]: “receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration;”**

775. This claim element is substantively identical to element [1c] in combination with claim 3 and would have been obvious for the reasons discussed above in connection with these claim elements.

- c. **[17h]: “wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file.”**

776. Brown discloses “wherein, responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of

the older version of the first file.”

777. Brown teaches that the server application (SFS) automatically transfers the modified file from the server to the second client device to replace an older version of the file. As discussed above with respect to claim [1c], it would have been obvious to update another device associated with the same user by transferring an updated version of a file received from one user device to the other. Synchronization in Brown’s system is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “Client machines synchronize with each other by synchronizing to a common server account.” *Id.*, 2:63-67. After the server account receives a modified file from one client device, the server application transfers that file to another client device associated with the common server account. *Id.*, 10:18-27, 10:53-55 (describing that when the “server 105 has changes that client 130 lacks,” then the client will “download[] all the pertinent changes from server 105”).

778. Brown expressly teaches that, **responsive to transferring the copy of the first file to the second client device, an older version of the first file stored on the second client device is automatically caused to be replaced with the copy of the first file such that the copy of the first file is stored on the second client device in lieu of the older version of the first file** if it exists: “On the other hand,

if less than a threshold number of message digests in the MDAs match, then *the entire file is downloaded from the server*. Once the download file is completely constructed, the client inserts the download file into its final location, *replacing an older version of the file if it exists.*” Brown, 13:18-23.

9. Independent Claim 19

779. Claim 19, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**non-transitory machine-readable media storing instructions**” instead of a “**system**.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 19 (media)
A system comprising:	One or more non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system, cause operations comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receiving, by the server system, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the	receiving, by the server system, from the first client device, first metadata associated with the version of the first

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;
determine that the server system is not in communication with a second client device associated with the user;	determining, by the server system, that the server system is not in communication with a second client device associated with the user;
store the copy of the first file on the server system;	storing, by the server system, the copy of the first file on the server system;
automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and	automatically transferring, by the server system, the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.	automatically transferring, by the server system, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to (i) resuming communication with the second client device and (ii) receiving the copy of the first file from the first client device.

780. The limitation reciting a “**non-transitory machine-readable media storing instructions that, when executed by one or more processors of a server system,**” cause operations would have been obvious for the reasons discussed above in connection with claim 13 of the ’823. Thus, for the reasons discussed above for

claim 1 and claim 13 of the '823, this claim would have been obvious.

10. Claim 20

781. Claim 20 depends from claim 12 and further recites “**wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.**” Claim 20 is not materially different from claim 3, as shown in the table below.

Claim 3 (system)	Claim 20 (media)
The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	The one or more non-transitory machine-readable media of claim 19, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

782. Because claim 20 presents no material difference from claim 3, claim

20 would have been obvious for the same reasons discussed above in connection with claim 3.

E. '622 Brown Grounds: Claims 1, 3-5, 7, 11, 12, 14, 16, and 17 of the '622 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink, Shappell, and Rao.

1. Independent Claim 1

783. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:

[1b] receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;

[1c] store the copy of the first file on the server system;

[1d] receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;

[1e] automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and

[1f] automatically transfer, over a network, the copy of the first file to the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

784. As shown in the table below, claim 1 of the '622 patent is similar to claim 1 of the '607 patent (substantively identical limitations appear on the same row).

'607 Claim 1	'622 Claim 1
1. A system comprising:	1. A system comprising:
a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:	a server system comprising one or more processors programmed with computer program instructions that, when executed, cause the server system to:
receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;	receive, over a network, a copy of a first file from a first client device associated with a user, wherein the copy of the first file is automatically received from the first client device responsive to the user modifying a content of the first file stored on the first client device, the copy of the first file being a version of the first file that is generated from the user modifying the content of the first file;
receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file;	receive, from the first client device, first metadata associated with the version of the first file that is generated from the user modifying the content of the first file, the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file; ²¹

²¹ The order in which this claim element appears in claim 1 of the '622 patent is altered in the table, but it is otherwise identical to a limitation that appears in claim 1 of the '607 patent. The fact that the limitation appears in a different order than the

determine that the server system is not in communication with a second client device associated with the user;	
store the copy of the first file on the server system;	store the copy of the first file on the server system;
automatically transfer the first metadata to the second client device based on the first priority being greater than the second priority such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and	automatically transfer, based on the first priority being greater than the second priority, the first metadata to the second client device such that the first metadata is transferred to the second client device prior to the copy of the first file being transferred to the second client device; and
automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to . . . receiving the copy of the first file from the first client device.	automatically transfer, over a network, the copy of the first file to the second client device associated with the user to replace an older version of the first file stored on the second client device, responsive to receiving the copy of the first file from the first client device.

785. For the reasons discussed above in connection with claim 1 of the '607 patent, claim 1 of the '622 patent would have been obvious.

2. Claim 3

786. This claim is substantively identical to claim 3 of the '607 and, for the reasons discussed above in connection with claim 3 of the '607 patent, would have been obvious.

identical limitation in the '607 patent does not change the required obviousness analysis that I performed above for the '607 patent and, therefore, the analysis presented in connection with the '607 patent for this limitation applies equally here for this limitation.

3. Claim 4

787. This claim is substantively identical to claim 4 of the '607 and, for the reasons discussed above in connection with claim 4 of the '607 patent, would have been obvious.

4. Claim 5

788. This claim is substantively identical to claim 5 of the '607 and, for the reasons discussed above in connection with claim 5 of the '607 patent, would have been obvious.

5. Claim 7

789. This claim is substantively identical to claim 7 of the '607 and, for the reasons discussed above in connection with claim 7 of the '607 patent, would have been obvious.

6. Independent Claim 11

790. Claim 11, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**” that performs the steps of the method. Thus, for the reasons discussed above in connection with claim 1, claim 11 would have been obvious.

7. Claim 12

791. Claim 12, which depends from claim 11, would have been obvious for the same reasons discussed above in connection with claim 11 and claim 1 of the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

'607 patent.

792. '622 element [12a] recites: **“initially determining, by the server system, that the server system is not in communication with the second client device associated with the user; and.”** This claim limitation would have been obvious for the reasons discussed in connection with '607 element [1d] and the reasons below.

793. For example, it would have been obvious to synchronize the server with a client device, for example the laptop computer described in Brown. The client would initiate communication with the server, and the server could respond based on a determination that the client and server are in communication. Also, it would have been obvious to use any handshake type function in TCP/IP or HTTP or any protocol, given that Brown teaches that TCP/IP and HTTP are used, or any protocol may be used: “The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. ***Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.***” Brown, 3:1-8. It would have been obvious in view of these disclosures to determine that the server is not in communication with

the client by, for example, initiating communications via the server and hearing no response from the client.

794. This element is also obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission to the offline client. Hesselink, ¶ 183 (“As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.***”); ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”).

795. Hesselink further discloses “automatically and repeatedly” checking whether synchronizing devices are in communication prior to initiating

synchronization: “As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.*** The persistence of the system ensures that updates are completed for all online computers, as if there is some failure or problem during the update, the connection restores itself to ensure that the update is completed.” Hesselink, ¶ 183; *id.*, ¶ 190 (“With this functionality, when the user works on a file, it is automatically updated in the remote device by the device module. Once the file has been put into the cache memory on the local computer, the user can work locally on the file via the user module, as described above. Therefore if the connection to the network is lost, this will not affect the user’s ability to work on the file. Updates to the official reference file (and optionally the file as located in other remote computers) obviously cannot be performed as long as the connection is broken. Furthermore, ***the system is persistent and may be configured on a user computer to automatically and repeatedly attempt to reconnect with any and all remote devices which are actively subscribed to when a connection is broken during an active subscription.*** The present software recognizes that the file is in cache and the user can continue

working locally when the connection is broken. When the network connection is restored, the software automatically updates the home file (official reference file) on the remote device via the local user module's transmission to the remote device module, as described above (and optionally, on any other computers where the file may reside). This feature is useful not only for when a network connection is accidentally broken, but in situations where the user moves to a location where the user knows a network connection is not possible, as in the example described above, while flying on an airplane, or in some other location that is not network accessible. Even in these situations, *as soon as the user logs into the network again, the official reference file may be automatically updated*, so that when the user returns to the location of the home file (official reference file), the home file is already up to date.”).

796. '622 element [12b] recites: “**automatically transferring, by the server system, the copy of the first file to the second client device to replace the older version of the first file stored on the second client device with the copy of the first file, responsive to (i) resuming communication with the second client device, (ii) determining that the server system is in communication with the second client device, (iii) and receiving the copy of the first file from the first**

client device.” This claim limitation would have been obvious for the reasons discussed in connection with ’607 element [1g] and the reasons below.

797. As discussed above in connection with element [12a], Hesselink discloses “automatically and repeatedly” checking whether synchronizing devices are in communication and resuming synchronization when previously offline devices come back online: “As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.*** The persistence of the system ensures that updates are completed for all online computers, as if there is some failure or problem during the update, the connection restores itself to ensure that the update is completed.” Hesselink, ¶ 183; *id.*, ¶ 190 (“With this functionality, when the user works on a file, it is automatically updated in the remote device by the device module. Once the file has been put into the cache memory on the local computer, the user can work locally on the file via the user module, as described above. Therefore if the connection to the network is lost, this will not affect the user’s ability to work on the file. Updates to the official reference file (and optionally the file as located in other remote computers) obviously cannot

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

be performed as long as the connection is broken. Furthermore, ***the system is persistent and may be configured on a user computer to automatically and repeatedly attempt to reconnect with any and all remote devices which are actively subscribed to when a connection is broken during an active subscription.*** The present software recognizes that the file is in cache and the user can continue working locally when the connection is broken. ***When the network connection is restored, the software automatically updates the home file (official reference file) on the remote device via the local user module's transmission to the remote device module, as described above (and optionally, on any other computers where the file may reside).*** This feature is useful not only for when a network connection is accidentally broken, but in situations where the user moves to a location where the user knows a network connection is not possible, as in the example described above, while flying on an airplane, or in some other location that is not network accessible. Even in these situations, ***as soon as the user logs into the network again, the official reference file may be automatically updated,*** so that when the user returns to the location of the home file (official reference file), the home file is already up to date.”). The remaining claim limitations would have been obvious for the reasons discussed above in connection with '607 element [1g].

798. The same motivations to combine Brown and Hesselink described

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

above for '823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Hesselink into Brown's systems and methods.

8. Claim 14

799. This claim is substantively identical to claim 14 of the '607 and, for the reasons discussed above in connection with claim 14 of the '607 patent, would have been obvious.

9. Independent Claim 16

800. Claim 16, for purposes of my analysis, is substantially similar to claim 1 except that it recites one or more **“non-transitory machine-readable media storing instructions”** instead of a **“system.”** For the reasons discussed above for claims 1 and 13 of the '823, this claim would have been obvious.

10. Claim 17

801. Claim 17 depends from claim 16 and further recites **“wherein at least one of the first computer system or the second computer system comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the first computer system or the second computer system assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.”** Claim 17 is not materially

different from claim 3 and, because claim 17 presents no material difference from claim 3, claim 17 would have been obvious for the same reasons discussed above in connection with claim 3.

Claim 3 (system)	Claim 17 (media)
3. The system of claim 1, wherein at least one of the server system or the first client device comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the server system or the first client device assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.	17. The one or more non-transitory machine-readable media of claim 16, wherein at least one of the first computer system or the second computer system comprises a priority assignment configuration to assign greater priority to metadata associated with files than priority assigned to the files such that at least one of the first computer system or the second computer system assigns the first priority to the first metadata and the second priority to the copy of the first file based on the priority assignment configuration.

F. '942 Ground 3: Claims 1, 3, 4, 10, 12 and 13 of the '942 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink Alone (Claims 1, 3, 4, 10, 12 and 13) and in Further Combination with Shappell (Claims 4 and 13).

1. Independent Claim 1

802. For reference, claim 1 recites:

[1p] A system comprising:

[1a] a first electronic device, associated with a user, configured to:

[1b] receive, via a first application at the first electronic device, a copy of a

modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;

[1c] determine whether the first electronic device is in communication with a third electronic device;

[1d] automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;

[1e] receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;

[1f] determine whether the first electronic device is in communication with the second electronic device; and

[1g] automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,

[1h] wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and

[1i] wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file

stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

a. [1p]: “A system comprising:”

803. The preamble of claim 1 recites “[a] system.” I understand that the preamble of a claim generally does not limit the claim, but to the extent that the preamble provides a claim limitation, it is disclosed by Brown, for the reasons discussed above in connection with ’823 claim [1p].

b. [1a]: “a first electronic device, associated with a user, configured to:”

804. Brown discloses a first electronic device, associated with a user.

805. For example, in Figure 1, an exemplary “**first electronic device**” is server **105**. The “server **105** includes server SFS **110**.” Brown, 3:56-57. “The server component is a server Synchronization File System (SFS) with a functional interface and metadata organization that enables efficient and reliable synchronization by the clients.” *Id.*, 2:59-62. “Typically, server SFS **110** is installed on a hard disk drive within server **105**” *Id.*, 3:59-60.

806. Brown’s server **105** is associated with a user, at least because the server SFS stores folders that are each associated with each user and can only be accessed by a given user once that user is logged in. Brown, 3:62-64 (“Stored on server SFS

110 are folders **115–1**, **115–2**, and **115–3** Each of the folders is assigned to a user. Once the user has logged in, the user can access the files and directories (cumulatively called directory entries) within the folder.”). Each user has a server “account.” *See id.*, 4:62-64 (one user account may be associated with multiple client devices). “Client machines synchronize with each other by synchronizing to a common server account.” *Id.*, 2:65-67. Changes to files are synchronized to “a user’s server account.” *Id.*, 5:3-4.

- c. **[1b]: “receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;”**

807. Brown in combination with Hesselink discloses receiving, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file.

808. For purposes of my analysis, the “**second electronic device associated with the user**” is disclosed by the client devices such as client **130** in Brown. Brown

discloses and renders obvious multiple client devices that are each associated with the same user. Brown addresses the problem in the prior art that an individual user may have multiple computers, such as work, home, and portable computers, and “needs to access the same files from the various computers”:

But as computers have become more affordable, people find themselves working with several machines. It is increasingly common for people to find themselves working on one machine at the office, a second machine at home, and having use of a portable computer when they need to have computer access while traveling. . . .

But with the increasing number of machines a person might find himself using comes an added complexity. Since a person typically accesses the same files from the various computers, the user needs to be certain that the files he is accessing are current.

Originally, people carried files on floppy disks from one machine to the next. But the increases in file size sometimes make floppy disks impractical. And if the user forgets to bring the files with him as he moves around, or forgets to move the latest versions of the files off the computer he most recently used, the user can find himself with several versions of the files, each of which contain desired portions.

Accordingly, a need remains for a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made, to address these and other problems associated with the prior art.

Brown, 1:24-50. Consistent with these teachings, Figure 1 specifically discloses two client computers, client **130** which is a user’s desktop computer, and client **135** which is the user’s portable computer: “Client **130** represents an ordinary desktop computer, as might be present at a user’s place of work or home. Portable computer

135 represents a notebook or laptop computer, as a user might take with him to a hotel room on business.” *Id.*, 4:20-24. In view of Brown’s teachings regarding the problem to be solved, a POSA would have understood that computers **130** and **135** are both, and obviously would both be, associated with the same user. Brown contemplates “multiple clients of a single user account” that may “push changes to the server concurrently” and that “the server SFS interleaves the changes.” *Id.*, 4:62-64. Therefore, “each user can have several clients accessing a single folder.” *Id.*, 11:55-56. Thus, Brown discloses and renders obvious multiple client devices associated with a single user.

809. Brown describes the synchronization of the client/server system of Figure 1 in Figures 4A-4C and 5 with reference to client **130**. Brown, Figs. 4A-4C, 10:7-9, 10:7-11:52. Brown further illustrates the synchronization of the client/server system of Figure 1 by describing detailed algorithms for client devices to synchronize with the server, including uploading and downloading metadata and files, in the flowcharts shown in Figures 8A-11F and corresponding text. *Id.*, Figs. 8A-11F, 13:57-16:39. Brown teaches that these flowcharts specifically correspond to the clients and server shown in Figure 1, namely client devices **130** and **135** and server **105**. *Id.*, 13:58-61 (“synchronizing the clients and server of FIG. 1”), 14:7-9, 14:60-62, 15:30-32. A POSA thus would have understood from Brown’s disclosures

that the disclosed synchronization process applies to changes to a file that are made on either of client devices **130** or **135** associated with the same user, which will then be synchronized to the other client device through the server. A POSA would have understood that the reference to the client devices associated with a user (such as devices **130** and **135**) are fungible in the sense that files would be synchronized bi-directionally between client devices **130** and **135** as files are modified on either of the devices.

810. Brown discloses or at least suggests that its server **105** (“**first electronic device**”) is configured to **receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file.**

811. Brown discloses and renders obvious that the **user modifies a content of an electronic file stored on the client device**, creating an “updated” version of the file. “Often, when files are **updated**, only a portion of a file is changed. For example, when a text document is edited, some paragraphs are removed, and other paragraphs are inserted.” Brown, 11:64-66.

812. Brown discloses and renders obvious that, responsive to the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

modification, and as part of the file synchronization process disclosed in Brown, the server receives a copy of the updated/modified file. “***When the client is uploading a file to the server, the client starts by making a copy of the file.*** The client SA uses the filter driver to read the file. The filter driver makes sure that the copy operation does not interfere with an application attempting to access the file during the copy. Copying the file is relatively quick, and ***once the copy is made the client SA can operate on the copy of the file*** without worrying about another application on the client trying to access the file.” Brown, 11:22-30. The copy of the updated file is received via the server’s “**first application,**” at the server (“SFS”). *Id.*, 2:59-62 (“The server component is a server Synchronization File System (SFS) with a functional interface and metadata organization that enables efficient and reliable synchronization by the clients.”).

813. Brown discloses that server 105 receives **a copy of the file** when more than a threshold portion of the file has changed. Brown teaches that either only the parts of the file that have changed or the entire file may be transferred, depending on the extent of the file changes. Specifically, Brown discloses that the “entire” modified file is uploaded to the server from the client device when more than a threshold portion of the file has changed. The threshold is used by separating the file into blocks and performing a hash function on each block and determining

message digest array (MDA) values. “MDAs are arrays of 16-byte message digests computed from each 4K block of a file.” Brown, 12:15-16. “Hash function **605** takes a block of the file, such as block **610** of file **615**, and computes the message digest, such as message digest **620** in MDA **625**. MDA **620** can then be used to determine if the file can be only partially uploaded.” *Id.*, 12:41-45. Then, “if less than a threshold number of message digests in the MDAs match, ***the entire file is transmitted.***” *Id.*, 12:49-51. *See also id.*, Fig. 11B (steps 1117, 1122, 1120), 15:41-54 (describing that “if a previous version of the file is on the server,” and if “an insufficient number of message digests match between the received and computed MDAs,” then at step 1120 the client “uploads the entire file”).

814. Brown discloses “automatic” synchronization of modified files. “From the user’s perspective, ***the synchronization process is automatic***, runs in the background and requires minimal monitoring or intervention. In an embodiment of the invention, the client SA initiates synchronization on a fixed time interval. But a person skilled in the art will recognize that ***the client SA can use any scheduling algorithm to initiate synchronization.*** In addition, the user can also initiate synchronization at any time manually.” Brown, 3:40-47. The **first electronic file** is received from a **second application at a second electronic device** (“client Synchronization Application (SA)”).

815. Brown discloses that the server's application (SFS) communicates with client applications (SAs) to facilitate file transfers between the server and the clients. Brown, 3:1-8 ("The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.").

816. As noted, in "an embodiment" of Brown, "the client SA initiates synchronization on a fixed time interval," but alternatively the system "can use any scheduling algorithm to initiate synchronization," and the user may manually initiate synchronization "at any time." Brown, 3:42-47. Once synchronization is initiated, a file that has been modified on a client device will be transferred to the server (where the entire file is transferred) as a result of that modification, as discussed previously and shown in Brown's flow chart figures. To the extent Brown were argued not to disclose the first client automatically transferring a copy of a file "responsive to the user modifying a content" of the file, Hesselink discloses and renders obvious that feature.

817. It would have been obvious in view of Brown in combination with Hesselink for the server described in Brown to automatically receive a copy of a modified file when the user modifies a content of the file.

818. Brown teaches that the Synchronization Application (“SA”) running on each of its client devices may monitor activity in the local client file system to identify changes to files. “The client SA monitors local file system activity within the directory (sometimes called a folder) on the client so that it can efficiently locate changes to send to the server SFS during synchronization.” Brown, 3:47-50. In particular, “filter driver **350** is used to monitor the activity of files within the folder on the client. Specifically, filter driver **350** watches for other applications accessing the files in folder **302**, so as to determine which files on the client have been changed.” *Id.*, 8:13-17.

819. It would have been obvious to implement the Brown system so that the server receives a copy of the modified file when a user modifies the file on a client device, rather than waiting for a fixed time interval. Brown teaches that the client Synchronization Application “can use any scheduling algorithm to initiate synchronization” and that the user can also initiate synchronization “at any time.” Brown also seeks to provide “*a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made.*” Brown, 1:46-

50. A POSA would have found it obvious to look to Hesselink for a technique to serve Brown's stated purpose to distribute files automatically across multiple clients to maintain currency at each client as changes are made.

820. Hesselink teaches a synchronization system where a given individual user may have multiple computers that synchronize versions of a given file, such as a home computer, work computer, and laptop computer, similar to Brown's disclosure that each user has multiple client devices for which files are synchronized. "The present system may also be optionally employed to ***perform synchronization of files among a user's multiple computers***. Referring back to the operations described above, the system is designed to ***automatically update*** at least the 'master copy' of a file at a remote location (i.e., the file at the location of the device module from where the file was accessed by the user module) ***as the user makes changes to the file locally*** on the computer associated with the user module." Hesselink, ¶ 210. "Thus, the user is ensured of having a completely up to date master copy of each shared data file at the home location/home locations (since home locations of shared files may vary). For example, the system may be set up such that when a user is away, working on his/her laptop, a personal file is automatically updated at the user's home computer storage device, while a business file is automatically updated at the user's work computer storage device." *Id.*, ¶ 192.

821. Hesselink teaches that when the user modifies or edits a file, the system automatically transmits the changes to the file over the network. The synchronization occurs “as soon as the update has been performed” at one device (for example, a home or office location), so that all connected computers “may be automatically updated concurrently with the updates of the device module file, each time an update is performed.” Hesselink, ¶¶ 183, 189. “***As the user works on the file***, such as by adding to the file, ***modifying or editing***, each time that the user saves his/her work, the system compares the differences between the saved version of the file and the previous version of the file, and, as long as the user module remains connected to the network, ***the changes/differences are transmitted over the network to the remote device module***, which are saved there, thereby maintaining the home file (i.e., official reference file, source file, master file or DataClient file) up to date with the latest revisions, as described above.” *Id.*, ¶ 189. “Optionally, all computers that are connected to the network, that share a file that the user is working on and that have a copy of that file in cache memory (or on a remote device) ***may be automatically updated concurrently with the updates of the device module file, each time an update is performed***, when such computers are actively subscribed, as noted above. Computers that are not actively subscribed or are offline can be automatically updated when reconnecting with the network or initiating an active

subscription.” *Id.*

822. Hesselink reiterates: “As noted above, ***when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.***” Hesselink, ¶ 183.

823. Hesselink further teaches that its disclosed technique of automatically updating copies of a file among a user’s multiple computers as the user makes changes to the file “may be set to apply to only one file or any number of files” at the user’s option: “The present system may also be optionally employed to perform synchronization of files among a user’s multiple computers. Referring back to the operations described above, the system is designed to automatically update at least the ‘master copy’ of a file at a remote location (i.e., the file at the location of the device module from where the file was accessed by the user module) as the user makes changes to the file locally on the computer associated with the user module. However, the user may be provided with alternative synchronization options by the present system. For example, the user may choose to always synchronize all copies of a file, wherein, whenever the user makes a change to that file locally, at any location, then all of the devices which subscribe to that file are automatically updated immediately. ***This option, and all others described below, may be set to apply to***

only one file or any number of files to which the user module is subscribed. At the device module end, such options may be chosen to apply in the same way. Another synchronization option provided by the present system is one in which a local copy of the file is stored on the local computer, and changes with at least one other remote device are automatically synched.” Hesselink, ¶ 210. The same motivations to combine Brown and Hesselink described above for ’823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Hesselink into Brown’s systems and methods.

d. [1c]: “determine whether the first electronic device is in communication with a third electronic device;”

824. Brown in combination with Hesselink discloses determining whether the first electronic device is in communication with a third electronic device.

825. For example, it would have been obvious to synchronize the server with a client device, for example the laptop computer (“**third electronic device**”) described in Brown. The client would initiate communication with the server, and the server could respond based on a determination of whether the client and server are in communication. Also, it would have been obvious to use any handshake type function in TCP/IP or HTTP or any protocol, given that Brown teaches that TCP/IP and HTTP are used, or any protocol may be used: “The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. ***Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.***” Brown, 3:1-8.

826. Hesselink discloses repeatedly checking whether other devices are in communication to synchronize, i.e., determining whether subscribed client devices are in communication with the server. Hesselink, ¶ 182; *id.*, ¶ 190 (“With this functionality, when the user works on a file, it is automatically updated in the remote device by the device module. Once the file has been put into the cache memory on the local computer, the user can work locally on the file via the user module, as described above. Therefore if the connection to the network is lost, this will not affect the user’s ability to work on the file. Updates to the official reference file (and optionally the file as located in other remote computers) obviously cannot be performed as long as the connection is broken. Furthermore, ***the system is persistent and may be configured on a user computer to automatically and repeatedly attempt to reconnect with any and all remote devices which are actively subscribed to when a connection is broken during an active subscription.*** The present software recognizes that the file is in cache and the user can continue working locally when the connection is broken. ***When the network connection is restored, the software***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

automatically updates the home file (official reference file) on the remote device via the local user module's transmission to the remote device module, as described above (and optionally, on any other computers where the file may reside). This feature is useful not only for when a network connection is accidentally broken, but in situations where the user moves to a location where the user knows a network connection is not possible, as in the example described above, while flying on an airplane, or in some other location that is not network accessible. Even in these situations, ***as soon as the user logs into the network again, the official reference file may be automatically updated,*** so that when the user returns to the location of the home file (official reference file), the home file is already up to date.”); Hesselink, ¶ 196 (“When updates are detected by a device module, the device module sends the update information over the network, optionally through a connection server and to all user modules that subscribe to each file that has been updated with regard to subscriptions.”).

827. This type of determination in Hesselink would have been obvious to use with Brown, to account for the situation in Brown where the second client device is not currently online. In the Brown system as obviously modified by Hesselink such that the modified file would be automatically transferred responsive to the modification from the first client through the server to the second client, when the

second client device is not currently online, it would have been obvious to determine whether or not the server is in communication with the second client, to determine whether or not the file could be transferred to the second client. Hesselink discloses: “As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.*** The persistence of the system ensures that updates are completed for all online computers, as if there is some failure or problem during the update, the connection restores itself to ensure that the update is completed.” Hesselink, ¶ 183.

- e. [1d]: “automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;”

828. Brown in combination with Hesselink discloses automatically sending, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to

receiving the modified first electronic file copy from the second electronic device.

829. For example, it would have been obvious to synchronize the server with a client device, for example the laptop client (“**third electronic device**”) described in Brown. The client would initiate communication with the server, and the server could respond based on a determination that the client and server are in communication, as discussed above for claim [1c]. Also, it would have been obvious to use any handshake type function in TCP/IP or HTTP or any protocol, given that Brown teaches that TCP/IP and HTTP are used, or any protocol may be used: “The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. *Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.*” Brown, 3:1-8.

830. Brown teaches that the synchronization is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server. *Client machines synchronize with each other by*

synchronizing to a common server account.” Id., 2:63-67.

831. A POSA would have understood and found it obvious that a **modified first electronic file copy** would have been **automatically** sent to the **third electronic device** (for example, laptop client **135**) from the server **responsive to receiving the modified first electronic file copy from the second electronic device**. Brown describes a synchronization process with respect to laptop client **135** that starts by checking whether the server has any updates for the user’s account, which would be updates from other client devices of that user such as desktop client **130**. Brown, 10:18-27 (“Otherwise, server 105 has changes that client 130 lacks. Server 105 then sends the SSD to client 130 (in response to a request for the SSD by the client), informing the client of the pertinent changes, as shown in box 415. Specifically, the SSD includes the SSI, the SIDs of any directories that contain changes since the last time client 130 synchronized with server 105, the SIDs of all items (files and directories) in the changed directories, and the metadata of all items (files and directories) that have been changed since the last time client 130 synchronized with server 105.”). Later, client **135** can send all pertinent changes made on client **135** to server **105** to update other client devices associated with the same user, as would be motivated by a user wanting to keep, for example, work and home computers synchronized as discussed in Brown. *Id.*, 10:53-55 (“Once client

130 has downloaded all the pertinent changes from server 105, client 130 can send all the pertinent changes made on client 130 to server 105.”).

832. Brown teaches that the server application (SFS) automatically transfers the modified file received from the **second electronic device** via the server to the **third electronic device** to replace an older version of the file. As discussed above with respect to ’823 claim [3c], it would have been obvious to update another device associated with the same user by transferring an updated version of a file received from one user device to the other. Synchronization in Brown’s system is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “Client machines synchronize with each other by synchronizing to a common server account.” *Id.* After the server account receives a modified file from one client device, the server application transfers that file to another client device associated with the common server account. *Id.*, 10:18-27, 10:53-55 (describing that when the “server 105 has changes that client 130 lacks,” then the client will “download[] all the pertinent changes from server 105”).

833. Brown expressly teaches that a downloaded file **replaces an older version of the file** if it exists: “On the other hand, if less than a threshold number of message digests in the MDAs match, then ***the entire file is downloaded from the***

server. Once the download file is completely constructed, the client inserts the download file into its final location, ***replacing an older version of the file if it exists.***” Brown, 13:18-23.

834. This element is also obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission to the offline client. Hesselink, ¶ 183 (“As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. Computers that are offline at such a time will be automatically updated as soon as those computers go online again.”); ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”).

835. In addition, a POSA would have found it obvious in view of Hesselink to use Brown’s server to automatically push updates from the server to connected

clients as soon as file updates occur. For example, Hesselink discloses: “As noted above, *when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.*” Computers that are offline at such a time will be automatically updated as soon as those computers go online again.” Hesselink, ¶ 183. A POSA would have been motivated to improve Brown’s system, to the extent it was deficient, with this improvement from Hesselink for the reasons discussed above in connection with claim 1 and, in particular, in view of Brown’s stated goal to provide “a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made” and Brown’s teaching that “a person skilled in the art [would] recognize that *the client SA can use any scheduling algorithm to initiate synchronization,*” including Hesselink’s algorithm for automatic updating. Brown, 1:46-50, 3:42-46.

- f. [1e]: “receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;”

836. Brown in combination with Hesselink discloses receiving, via the first

application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file.

837. Brown teaches that synchronization is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server. ***Client machines synchronize with each other by synchronizing to a common server account.***” *Id.*, 2:63-67. As discussed above for ’823 claim [1b], a POSA thus would have understood from Brown’s disclosures that the disclosed synchronization process applies to changes to a file that are made on either of client devices **130** or **135** associated with the same user, which will then be synchronized to the other client device through the server. A POSA would have understood that the reference to the client devices associated with a user (such as devices **130** and **135**) are fungible in the sense that files would be synchronized bi-directionally between client devices **130** and **135** as files are modified on either of the devices.

838. A POSA would have understood and found it obvious that a second file on the second client device (for example, laptop computer **135**) would be received

via the second client's application to the server application when that file is modified. Brown describes a synchronization process with respect to desktop client **130** that starts by checking whether the server has any updates for the user's account, which would be updates from other client devices of that user such as laptop client **135**. Brown, 10:18-27 ("Otherwise, server 105 has changes that client 130 lacks. Server 105 then sends the SSD to client 130 (in response to a request for the SSD by the client), informing the client of the pertinent changes, as shown in box 415. Specifically, the SSD includes the SSI, the SIDs of any directories that contain changes since the last time client 130 synchronized with server 105, the SIDs of all items (files and directories) in the changed directories, and the metadata of all items (files and directories) that have been changed since the last time client 130 synchronized with server 105."). Later, client **130** can send all pertinent changes made on client **130** to server **105** to update other client devices associated with the same user, as would be motivated by a user wanting to keep, for example, work and home computers synchronized as discussed in Brown. *Id.*, 10:53-55 ("Once client 130 has downloaded all the pertinent changes from server 105, client 130 can send all the pertinent changes made on client 130 to server 105."). Thus, this limitation would have been obvious for the same reasons discussed above in connection with claim [1b].

- g. [1f]: “determine whether the first electronic device is in communication with the second electronic device; and”

839. Brown in combination with Hesselink discloses or renders obvious determining whether the first electronic device is in communication with the second electronic device.

840. For the reasons discussed above, Brown’s “clients” are fungible, so it would have been obvious to implement the functionality described in claim [1c] for determining whether the first electronic device is in communication with the third electronic device but instead for “**determin[ing] whether the first electronic device is in communication with the second electronic device.**” In other words, each of Brown’s clients undergoes the same synchronization process, including determinations related to communication between a client and server, as discussed above for ’823 claim 1. Thus, for the same reasons above for claim [1c], Brown discloses that its server may detect whether any of its clients are online or offline and in communication or out of communication with the server. In addition, this element would have been obvious in view of Brown in combination with Hesselink for the reasons above for claim [1c].

- h. [1g]: “**automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is**

**in communication with the second electronic device
and responsive to receiving the modified second
electronic file copy from the third electronic device,”**

841. Brown in combination with Hesselink discloses automatically sending, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device.

842. For the reasons discussed above, Brown’s “clients” are fungible. Thus, for the reasons discussed above in connection with claims [1c] and [1d], it would have been obvious to automatically transfer the modified “**second electronic file copy**” from Brown’s server, via the server’s “**first application,**” to the “**second application**” “**responsive to the determination that the first electronic device**” (server) “**is in communication with the second electronic device**” (desktop client 130) “**and responsive to receiving the modified second electronic file copy from the third electronic device**” (laptop client 135).

- i. **[1h]: “wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and”**

843. Brown discloses “wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file.”

844. Brown teaches that the server application (SFS) automatically transfers the modified first file received from the second electronic device to the third electronic device to replace an older version of the file. As discussed above, it would have been obvious to update another device associated with the same user by transferring an updated version of a file received from one user device to the other. Synchronization in Brown’s system is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “Client machines synchronize with each other by synchronizing to a common server account.” *Id.* After the server account receives a modified file

from one client device, the server application transfers that file to another client device associated with the common server account. *Id.*, 10:18-27, 10:53-55 (describing that when the “server 105 has changes that client 130 lacks,” then the client will “download[] all the pertinent changes from server 105”).

845. Brown expressly teaches that a downloaded file replaces an older version of the file if it exists: “On the other hand, if less than a threshold number of message digests in the MDAs match, then ***the entire file is downloaded from the server.*** Once the download file is completely constructed, the client inserts the download file into its final location, ***replacing an older version of the file if it exists.***” Brown, 13:18-23.

- j. [1i]: “wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.”

846. Brown discloses wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second

electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

847. For the reasons discussed above, Brown’s “clients” are fungible. Thus, for the reasons discussed above in connection with claim [1h], it would have been obvious to implement this limitation with respect to a **modified second electronic file copy** sent to the **second electronic device**.

2. **Claim 3: “The system of claim 1, wherein the first electronic device is configured to: store, via the first application, the modified first electronic file copy to a memory device associated with the first electronic device.”**

848. Brown discloses wherein the first electronic device is configured to: store, via the first application, the modified first electronic file copy to a memory device associated with the first electronic device.

849. Brown discloses that the server SFS stores the modified file in memory. “FIG. 1 shows a server with several clients accessing files stored on a server SFS on the server, according to an embodiment of the invention. In FIG. 1, server **105** includes server SFS **110**. Server **105** includes all the typical elements of a server, such as a central processor, *memory*, bus, disk space, etc. Typically, server SFS **110** is installed on a hard disk drive within server **105**, but a person skilled in the art will recognize other forms of media usable by server **105**: for example, removable media or optical media. *Stored on server* SFS **110** are folders **115**—

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

1, 115–2, and 115–3. Although FIG. 1 shows only three such folders, a person skilled in the art will recognize that there can be more or fewer folders. Each of the folders is assigned to a user. Once the user has logged in, the user can access the files and directories (cumulatively called directory entries) within the folder. For example, folder 115–1 is shown with three files 117–1, 117–2, and 117–3.” Brown, 3:54-4:3. “For file uploads, a file is first uploaded to a temporary file on the server. Then, *after the file data is on the server, the server SFS inserts the file and its metadata into the user’s server account database* in an atomic operation.” *Id.*, 4:64-5:1. “Once the file has been completely uploaded, the server then stores the file in the specified directory and updates the file metadata.” *Id.*, 13:1-3.

3. **Claim 4: “The system of claim 3, wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.”**

850. Brown in combination with Shappell discloses wherein the modified first electronic file copy is stored on the memory device associated with the first electronic device in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.

851. As discussed above in connection with claim 3, Brown discloses that **“the modified first electronic file copy is stored on the memory device associated**

with the first electronic device.”

852. Brown discloses and renders obvious that the server SFS will store the modified file in memory in lieu of an older version of the file that is “replaced” by the modified file. Brown teaches: “Access to the server SFS is restricted to file level operations via the protocol where new files can be uploaded, **existing files can be** downloaded, **replaced**, renamed, moved, or deleted, but an existing file cannot be modified directly.” Brown, 4:44-47. A POSA would have found it obvious in view of this disclosure to replace old versions of files with new ones at the server when instructed to do so via the aforementioned file level operation. Motivation to do so would have included the conservation of memory storage resources via the avoidance of storing an endless set of old versions of files.

853. In addition, a POSA would have found it obvious that where a client uploads a modified version of a file to the server, the teaching that “existing files can be . . . replaced” on the server would obviously mean that an older version of the file on the server would be replaced by the uploaded modified file, which would be stored on the server in place of the older version. Brown describes that when the client uploads a modified file to the server, the server checks for “the version of the file on the server” and then uploads the file (which will upload the complete file, unless a sufficient number of MDA values match). Brown, 12:44-13:3. The server

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

then “constructs the new version of the file” and “updates the file metadata.” *Id.*, 12:65-13:3. Brown states: “Before the client SA uploads a file, it computes an MDA from the file. It then requests from the server the MDA for ***the version of the file on the server*** by sending to the server the SID of the file, the name of the file, and the directory to which the file is to be uploaded. The server then checks to see if it has a file with that SID or if there is a file with the same name as that specified by the client in the directory to which the client is uploading the file. If the server finds a version of the file, it returns the file’s MDA to the client. The client SA compares the two MDAs and if a sufficiently high number of message digests match, it performs a special upload where only the differing message digests and their corresponding 4K data blocks are uploaded. ***The server constructs the new version of the file*** by starting with a copy of the previous version and modifying it with the uploaded data. Once the file has been completely uploaded, ***the server then stores the file in the specified directory and updates the file metadata.***” *Id.*, 12:53-13:3. A POSA would have found it obvious that this operation is an example of where “existing files can be . . . replaced” on the server.

854. Furthermore, Brown’s discussion of synchronization would have motivated a POSA to consider the problem of a user having copies of the same file on their home computer, work computer, and the server and the issue of

synchronizing all three devices to make sure they are updated with the most recent updates on each respective version. For example, Brown expressly references a client's desktop computer and laptop computer as two exemplary "clients," and describes bidirectional synchronization between such devices via a common server account: "Client 130 represents an ordinary desktop computer, as might be present at a user's place of work or home. Portable computer 135 represents a notebook or laptop computer, as a user might take with him to a hotel room on business." Brown, 4:20-24; *id.*, 1:62-63 ("Client machines synchronize with each other by synchronizing to a common server account.").

855. A POSA would have found it obvious that in this type of use case, after the copy of a file stored on the user's desktop computer (which may be called "Version 1") is transmitted from the user's desktop computer to the user's laptop computer via the server, the user may continue to edit that document on the laptop computer while the user is on a business trip. Based on Brown's bidirectional file synchronization architecture, a copy of the updated, edited file at the laptop computer (call it "Version 2") would then be transmitted from the laptop computer, through the server, and back to the user's desktop computer. A POSA would have been motivated to look for solutions to replace old versions of files with new ones in this use case to conserve memory storage resources. A POSA would have found

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

it obvious that one possibility would be for the updated modified file (“Version 2”) to replace the outdated older version of the file (“Version 1”) on the server and the desktop computer. A POSA would have found it obvious to replace the older version of the file for reasons including the obvious benefit of preventing copies of outdated versions from endlessly accumulating in storage on the user’s computers.

856. Although claim 4 would have been obvious in view of Brown alone, it is further obvious in view of Brown in combination with Shappell, which expressly teaches and motivates this feature with its “automatic replication” option, as discussed further below.

857. Shappell’s “automatic replication” option further renders obvious implementing Brown’s server **105 wherein the modified first electronic file copy is stored on the memory device associated with server 105 (first electronic device) in lieu of an older version of the first electronic file copy that was stored on the memory device associated with the first electronic device.** Shappell teaches a user-selectable “File Replication” setting where the user can switch between (1) manually downloading files “on an on-demand basis,” and (2) automatically downloading files when available, with an “automatic replication” setting. Shappell, ¶¶ 62, 81. When “automatic replication” is enabled, Shappell teaches automatically replacing an “out-of-date file” with an “updated version” on a

receiving computer. *Id.*, ¶ 81 (“When a modification of a file contained in a shared space occurs, a notification to other members in the group occurs (see steps 1902, 1904 and 1906 in FIG. 19). In a preferred embodiment, upon receipt of such notification, member machines that have previously obtained a copy of the shared file will ***remove the local copy of the out-of-date file. . . . [I]f automatic replication of the shared file is enabled, the local machine obtains an updated version of the file***, as is shown in step 2006 in FIG. 20.”), *id.*, Figs. 19-20.

858. It would have been obvious that, when implementing Shappell’s “automatic replication” feature, server **105** should delete all old versions of a file stored on server **105** after receiving an updated version of the file, given that no user device would ever want or need an older version of the file (they are deleted on the client devices). A POSA would have been motivated to incorporate Shappell’s auto-replication to address Brown’s aim to provide “a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made.” Brown, 1:46-50.

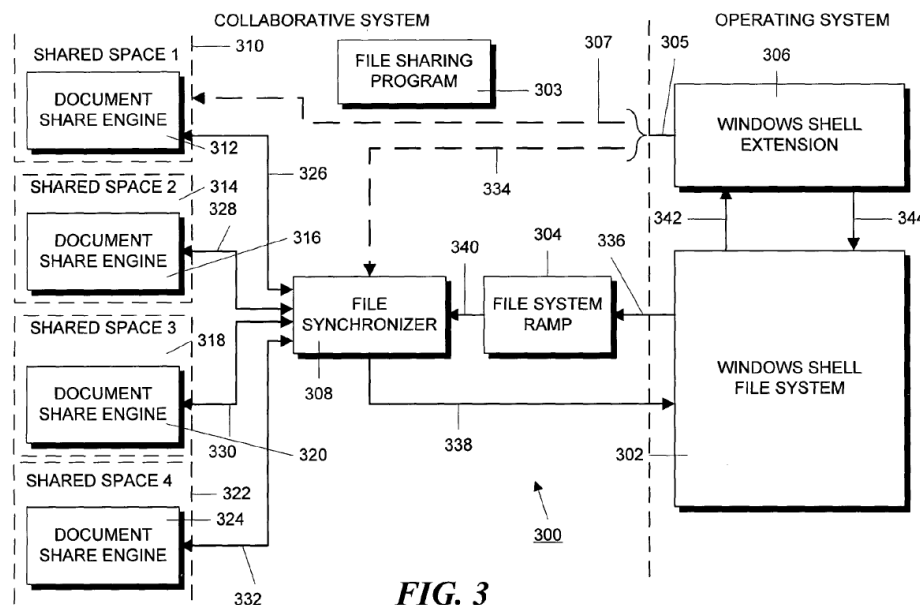
859. A POSA would have further been motivated to replace an older version of a file on server **105** with a newer version in view of Moromisato, which discloses a server-based system in which a change to the server data automatically causes the server to send a command to all clients to modify their data. As part of this process,

Moromisato discloses that an older version of a file is replaced on the server:
“Alternatively, in a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. Each collaborator that wants to modify the data sends information to the server to effect a change in the server data. ***The server modifies its copy of the data and then transmits the command to all collaborators.*** Each collaborator uses the command to update its local data copy. . . . In a server-based collaborative system in which all files are kept on the server, the file to be synchronized will already be available to the user. However in server-based collaborative systems that use local data copies and in distributed collaborative systems, that provide asymmetric file service as discussed above, ***the file may not be present on the local data store and must be downloaded to the local data store from an appropriate source. In the case of server-based systems, that source will be the server.***” Moromisato, 1:47-54, 14:47-55.

860. Moromisato further teaches: “File synchronizer 308 supports automatic and manual synchronization of changes between the synchronized file system folder and the collaborative system ‘sites’. ***The automatic synchronization mode requires that changes to a file be synchronized as they happen.*** . . . Once the file synchronizer 308 determines that the file system has been changed it passes the

changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may either disseminate the entire updated file contents*** or determine the changes made to the file by comparing the current file contents against a copy of the contents that it maintains and disseminate only the differences.” Moromisato, 19:49-53, 20:8-16.

861. Like Brown, Moromisato describes embodiments that use a “server based” system to implement file synchronization functionality. Moromisato, 1:47-57 (“[I]n a server based system, in addition to the data copy maintained in the server, additional local data copies may be maintained in each collaborating computer. ***Each collaborator that wants to modify the data sends information to the server to effect a change in the server data.***”). A POSA considering Brown’s similar server-based embodiments would have been motivated to consider the design of Moromisato’s file synchronizer illustrated in Figure 3:



Id., Fig. 3; *see also id.*, 7:25-28 (“The file synchronizer 308 is a software program that insures that any changes made to a file in the Windows shell file system 302 are also made to copies of that file that are stored in the collaborative system and vice versa.”).

862. Moromisato expressly discloses using the “file synchronizer 308” shown in Figure 3 that is configured to automatically transfer a modified file copy to a user device (i.e., any one of the devices running a respective “shared space” shown in Figure 3). Moromisato, 20:8-16; *id.*, 6:42-44 (“Each shared space is an instantiation of one or more activities operable on each of the collaborating computers of members of that shared space.”). Moromisato’s file synchronizer

replaces an older version of the file stored on the device with a modified copy from another device having the content modified by a user. *Id.*, 20:8-16 (“Once the file synchronizer 308 determines that the file system has been changed it passes the changes to the document share engine in the shared workspace which thereupon disseminates the changes. ***If a file has been edited, then the document share engine may . . . disseminate the entire updated file contents . . .***”); see also *id.*, 21:19 (listing “[f]ile edit (replace)” as a file-change command that the file synchronizer would be tasked with processing). It would have been obvious in view of these disclosures to replace an older version of a file on server **105** with a newer version received from a client. It would have been obvious to combine Brown and Hesselink with Shappell for the reasons discussed above for ’823 Ground 3, claim 1.

4. Independent Claim 10

863. Claim 10, for purposes of my analysis, is substantially similar to claim 1 except that it recites a “**method**” instead of a “**system**.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 10 (system)
A system comprising:	A method comprising:
a first electronic device, associated with a user, configured to:	

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

receive, via a first application at the first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with the user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;	receiving, via a first application at a first electronic device, a copy of a modified first electronic file from a second application at a second electronic device associated with a user, wherein the modified first electronic file copy is automatically received from the second application responsive to the user modifying a content of the first electronic file;
determine whether the first electronic device is in communication with a third electronic device;	determining whether the first electronic device is in communication with a third electronic device;
automatically send, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;	automatically sending, via the first application, the modified first electronic file copy to a third application at the third electronic device responsive to the determination that the first electronic device is in communication with the third electronic device and responsive to receiving the modified first electronic file copy from the second electronic device;
receive, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;	receiving, via the first application, a copy of a modified second electronic file from the third application at the third electronic device associated with the user, wherein the modified second electronic file copy is automatically received from the third application responsive to the user modifying a content of the second electronic file;
determine whether the first electronic device is in communication with the second electronic device; and	determining whether the first electronic device is in communication with the second electronic device; and

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

automatically send, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,	automatically sending, via the first application, the modified second electronic file copy to the second application at the second electronic device responsive to the determination that the first electronic device is in communication with the second electronic device and responsive to receiving the modified second electronic file copy from the third electronic device,
wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and	wherein, responsive to sending the modified first electronic file copy to the third electronic device, an older version of the first electronic file stored on the third electronic device is automatically caused to be replaced with the modified first electronic file copy such that the modified first electronic file copy is stored on the third electronic device in lieu of the older version of the first electronic file, and
wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.	wherein, responsive to sending the modified second electronic file copy to the second electronic device, an older version of the second electronic file stored on the second electronic device is automatically caused to be replaced with the modified second electronic file copy such that the modified second electronic file copy is stored on the second electronic device in lieu of the older version of the second electronic file.

864. Because claim 10 presents no material difference from claim 1, claim 10 would have been obvious for the same reasons discussed above in connection

with claim 1.

5. Claim 12

865. This claim is substantively identical to claim 3 and, for the reasons discussed above in connection with claim 3, would have been obvious.

6. Claim 13

866. This claim is substantively identical to claim 4 and, for the reasons discussed above in connection with claim 4, would have been obvious.

G. '561 Ground 3: Claims 1, 3, 4, 8, 10 and 11 of the '561 Patent Would Have Been Obvious in View of Brown in Combination with Hesselink.

1. Independent Claim 1

867. For reference, claim 1 recites:

[1p] A system, comprising:

[1a] a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:

[1b] receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and

[1c] wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and

[1d] wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and

[1e] automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;

[1f] wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.

b. [1p]: “A system comprising:”

868. The preamble of claim 1 recites “[a] system.” I understand that the preamble of a claim generally does not limit the claim, but to the extent that the preamble provides a claim limitation, it is disclosed by Brown, for the reasons discussed above in connection with ’823 claim [1p].

c. [1a]: “a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:”

869. Brown discloses a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user.

870. For example, the embodiment in Figure 1 of Brown discloses an exemplary “**first electronic device**” in the form of server **105**. The “**first application**” is server Synchronization File System (SFS), disclosed by SFS **110**. “The server component is a server Synchronization File System (SFS) with a functional interface and metadata organization that enables efficient and reliable synchronization by the clients.” Brown, 2:59-62. “Typically, server SFS **110** is installed on a hard disk drive within server **105**” *Id.*, 3:59-60.

871. For purposes of my analysis, an exemplary “**second electronic device**” is client **130** in Brown and an exemplary “**third electronic device**” is client **135** in Brown. Brown discloses and renders obvious multiple client devices that are each associated with the same user. Brown addresses the problem in the prior art that an individual user may have multiple computers, such as work, home, and portable computers, and “needs to access the same files from the various computers”:

But as computers have become more affordable, people find themselves working with several machines. It is increasingly common for people to find themselves working on one machine at the office, a second machine at home, and having use of a portable computer when they need to have computer access while traveling. . . .

But with the increasing number of machines a person might find himself using comes an added complexity. Since a person typically accesses the same files from the various computers, the user needs to be certain that the files he is accessing are current.

Originally, people carried files on floppy disks from one machine to the

next. But the increases in file size sometimes make floppy disks impractical. And if the user forgets to bring the files with him as he moves around, or forgets to move the latest versions of the files off the computer he most recently used, the user can find himself with several versions of the files, each of which contain desired portions.

Accordingly, a need remains for a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made, to address these and other problems associated with the prior art.

Brown, 1:24-50. Consistent with these teachings, Figure 1 specifically discloses two client computers, client **130** which is a user's desktop computer, and client **135** which is the user's portable computer: "Client **130** represents an ordinary desktop computer, as might be present at a user's place of work or home. Portable computer **135** represents a notebook or laptop computer, as a user might take with him to a hotel room on business." *Id.*, 4:20-24. In view of Brown's teachings regarding the problem to be solved, a POSA would have understood that computers **130** and **135** are both, and obviously would both be, associated with the same user. Brown contemplates "multiple clients of a single user account" that may "push changes to the server concurrently" and that "the server SFS interleaves the changes." *Id.*, 4:62-64. Therefore, "each user can have several clients accessing a single folder." *Id.*, 11:55-56. Thus, Brown discloses and renders obvious multiple client devices associated with a single user.

872. Brown's server **105** is associated with a user, at least because the server

SFS stores folders that are each associated with each user and can only be accessed by a given user once that user is logged in. Brown, 3:62-4:2 (“Stored on server SFS **110** are folders **115–1**, **115–2**, and **115–3** Each of the folders is assigned to a user. Once the user has logged in, the user can access the files and directories (cumulatively called directory entries) within the folder.”). Each user has a server “account.” *See id.*, 4:62-64 (one user account may be associated with multiple client devices). “Client machines synchronize with each other by synchronizing to a common server account.” *Id.*, 2:65-67. Changes to files are synchronized to “a user’s server account.” *Id.*, 5:3-4.

873. The server **105** is “**in communication with**” the clients **130** and **135** via their respective Synchronization Applications (SAs), as illustrated in Figure 1. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server.” Brown, 2:63-65. “Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.” *Id.*, 3:6-8.

874. **Selective execution** of the **first application** may be implemented based on, for example, a determination as to whether any client device has initiated communication for purposes of synchronization. Brown, 3:6-8 (“Communication between the client SA and the server SFS is initiated by the client SA and responded

to by the server SFS.”). In other words, the server may selectively execute the SFS instructions only when necessary to service client synchronization requests (e.g., when clients have file updates). At a minimum, selective execution would have been obvious in view of these disclosures in combination with the knowledge of a POSA. For example, a POSA would understand that any computer processor selectively executes application instructions by switching back and forth between application code and operating system code. As another example, an application is selectively executed only when necessary and when called upon to execute certain functionality, such as the functionality recited in this claim limitation.

- d. **[1b]: “receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and”**

875. Brown in combination with Hesselink discloses receiving from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file.

876. Brown discloses and renders obvious that the server **105** is configured to receive from client **130**’s SA (“**a second application executable on the second electronic device**”) a copy of a first electronic file automatically transferred when

the user modifies a content of the file.

877. Brown describes the synchronization of the client/server system of Figure 1 in Figures 4A-4C and 5 with reference to client **130**. Brown, Figs. 4A-4C, 10:7-9, 10:7-11:52. Brown further illustrates the synchronization of the client/server system of Figure 1 by describing detailed algorithms for client devices to synchronize with the server, including uploading and downloading metadata and files, in the flowcharts shown in Figures 8A-11F and corresponding text. *Id.*, Figs. 8A-11F, 13:57-16:39. Brown teaches that these flowcharts specifically correspond to the clients and server shown in Figure 1, namely client devices **130** and **135** and server **105**. *Id.*, 13:58-61 (“synchronizing the clients and server of FIG. 1”), 14:7-9, 14:60-62, 15:30-32. A POSA thus would have understood from Brown’s disclosures that the disclosed synchronization process applies to changes to a file that are made on either of client devices **130** or **135** associated with the same user, which will then be synchronized to the other client device through the server. A POSA would have understood that the reference to the client devices associated with a user (such as devices **130** and **135**) are fungible in the sense that files would be synchronized bi-directionally between client devices **130** and **135** as files are modified on either of the devices.

878. Brown discloses or at least suggests that its server **105** (“**first electronic**

device”) is configured to **receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file.**

879. Brown discloses and renders obvious that the **user modifies a content of the first electronic file**, creating an “updated” version of the file. “Often, when files are **updated**, only a portion of a file is changed. For example, when a text document is edited, some paragraphs are removed, and other paragraphs are inserted.” Brown, 11:64-66.

880. Brown discloses and renders obvious that, responsive to the modification, and as part of the file synchronization process disclosed in Brown, the server receives a copy of the updated/modified file. “***When the client is uploading a file to the server, the client starts by making a copy of the file.*** The client SA uses the filter driver to read the file. The filter driver makes sure that the copy operation does not interfere with an application attempting to access the file during the copy. Copying the file is relatively quick, and ***once the copy is made the client SA can operate on the copy of the file*** without worrying about another application on the client trying to access the file.” Brown, 11:22-30. The copy of the updated file is received via the server’s “**first application**,” at the server (“SFS”). *Id.*, 2:59-62

(“The server component is a server Synchronization File System (SFS) with a functional interface and metadata organization that enables efficient and reliable synchronization by the clients.”).

881. Brown discloses that server 105 receives **a copy of the file** when more than a threshold portion of the file has changed. Brown teaches that either only the parts of the file that have changed or the entire file may be transferred, depending on the extent of the file changes. Specifically, Brown discloses that the “entire” modified file is uploaded to the server from the client device when more than a threshold portion of the file has changed. The threshold is used by separating the file into blocks and performing a hash function on each block and determining message digest array (MDA) values. “MDAs are arrays of 16-byte message digests computed from each 4K block of a file.” Brown, 12:15-16. “Hash function **605** takes a block of the file, such as block **610** of file **615**, and computes the message digest, such as message digest **620** in MDA **625**. MDA **620** can then be used to determine if the file can be only partially uploaded.” *Id.*, 12:41-45. Then, “if less than a threshold number of message digests in the MDAs match, ***the entire file is transmitted.***” *Id.*, 12:49-51. *See also id.*, Fig. 11B (steps 1117, 1122, 1120) 15:41-54 (describing that “if a previous version of the file is on the server,” and if “an insufficient number of message digests match between the received and

computed MDAs,” then at step 1120 the client “uploads the entire file”).

882. Brown discloses “automatic” synchronization of modified files. “From the user’s perspective, *the synchronization process is automatic*, runs in the background and requires minimal monitoring or intervention. In an embodiment of the invention, the client SA initiates synchronization on a fixed time interval. But a person skilled in the art will recognize that *the client SA can use any scheduling algorithm to initiate synchronization*. In addition, the user can also initiate synchronization at any time manually.” Brown, 3:40-47. The **first electronic file** is received from a **second application at a second electronic device** (“client Synchronization Application (SA)”).

883. Brown discloses that the server’s application (SFS) communicates with client applications (SAs) (executable on the respective clients) to facilitate file transfers between the server and the clients. Brown, 3:1-8 (“The client SA communicates with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.”).

884. As noted, in “an embodiment” of Brown, “the client SA initiates

synchronization on a fixed time interval,” but alternatively the system “can use any scheduling algorithm to initiate synchronization,” and the user may manually initiate synchronization “at any time.” Brown, 3:42-47. Once synchronization is initiated, a file that has been modified on a client device will be transferred to the server (where the entire file is transferred) as a result of that modification, as discussed previously and shown in Brown’s flow chart figures. To the extent Brown were argued not to disclose the first client automatically transferring a copy of a file “responsive to the user modifying a content” of the file, Hesslink discloses and renders obvious that feature.

885. It would have been obvious in view of Brown in combination with Hesslink for the server described in Brown to automatically receive a copy of a modified file when the user modifies a content of the file.

886. Brown teaches that the Synchronization Application (“SA”) executable and running on each of its client devices may monitor activity in the local client file system to identify changes to files. “The client SA monitors local file system activity within the directory (sometimes called a folder) on the client so that it can efficiently locate changes to send to the server SFS during synchronization.” Brown, 3:47-50. In particular, “filter driver **350** is used to monitor the activity of files within the folder on the client. Specifically, filter driver **350** watches for other applications

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

accessing the files in folder **302**, so as to determine which files on the client have been changed.” *Id.*, 8:13-17.

887. It would have been obvious to implement the Brown system so that the server receives a copy of the modified file when a user modifies the file on a client device, rather than waiting for a fixed time interval. Brown teaches that the client Synchronization Application “can use any scheduling algorithm to initiate synchronization” and that the user can also initiate synchronization “at any time.” Brown also seeks to provide “***a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made.***” Brown, 1:46-50. A POSA would have found it obvious to look to Hesselink for a technique to serve Brown’s stated purpose to distribute files automatically across multiple clients to maintain currency at each client as changes are made.

888. Hesselink teaches a synchronization system where a given individual user may have multiple computers that synchronize versions of a given file, such as a home computer, work computer, and laptop computer, similar to Brown’s disclosure that each user has multiple client devices for which files are synchronized. “The present system may also be optionally employed to ***perform synchronization of files among a user’s multiple computers.*** Referring back to the operations described above, the system is designed to ***automatically update*** at least the ‘master

copy’ of a file at a remote location (i.e., the file at the location of the device module from where the file was accessed by the user module) ***as the user makes changes to the file locally*** on the computer associated with the user module.” Hesselink, ¶ 210. “Thus, the user is ensured of having a completely up to date master copy of each shared data file at the home location/home locations (since home locations of shared files may vary). For example, the system may be set up such that when a user is away, working on his/her laptop, a personal file is automatically updated at the user’s home computer storage device, while a business file is automatically updated at the user’s work computer storage device.” *Id.*, ¶ 192.

889. Hesselink teaches that when the user modifies or edits a file, the system automatically transmits the changes to the file over the network. The synchronization occurs “as soon as the update has been performed” at one device (for example, a home or office location), so that all connected computers “may be automatically updated concurrently with the updates of the device module file, each time an update is performed.” Hesselink, ¶¶ 183, 189. “***As the user works on the file***, such as by adding to the file, ***modifying or editing***, each time that the user saves his/her work, the system compares the differences between the saved version of the file and the previous version of the file, and, as long as the user module remains connected to the network, ***the changes/differences are transmitted over the network***

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

to the remote device module, which are saved there, thereby maintaining the home file (i.e., official reference file, source file, master file or DataClient file) up to date with the latest revisions, as described above.” *Id.*, ¶ 189. “Optionally, all computers that are connected to the network, that share a file that the user is working on and that have a copy of that file in cache memory (or on a remote device) ***may be automatically updated concurrently with the updates of the device module file, each time an update is performed***, when such computers are actively subscribed, as noted above. Computers that are not actively subscribed or are offline can be automatically updated when reconnecting with the network or initiating an active subscription.” *Id.*

890. Hesselink reiterates: “As noted above, ***when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.***” Hesselink, ¶ 183.

891. Hesselink further teaches that its disclosed technique of automatically updating copies of a file among a user’s multiple computers as the user makes changes to the file “may be set to apply to only one file or any number of files” at the user’s option: “The present system may also be optionally employed to perform synchronization of files among a user’s multiple computers. Referring back to the

operations described above, the system is designed to automatically update at least the ‘master copy’ of a file at a remote location (i.e., the file at the location of the device module from where the file was accessed by the user module) as the user makes changes to the file locally on the computer associated with the user module. However, the user may be provided with alternative synchronization options by the present system. For example, the user may choose to always synchronize all copies of a file, wherein, whenever the user makes a change to that file locally, at any location, then all of the devices which subscribe to that file are automatically updated immediately. ***This option, and all others described below, may be set to apply to only one file or any number of files*** to which the user module is subscribed. At the device module end, such options may be chosen to apply in the same way. Another synchronization option provided by the present system is one in which a local copy of the file is stored on the local computer, and changes with at least one other remote device are automatically synched.” Hesselink, ¶ 210. The same motivations to combine Brown and Hesselink described above for ’823 Ground 1 apply equally here and, for such reasons, it would have been obvious to incorporate the above teachings from Hesselink into Brown’s systems and methods.

- e. **[1c]: “wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file; and”**

892. Brown discloses wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred from the third application when the user modifies a content of the second electronic file.

893. Brown teaches that synchronization is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server. *Client machines synchronize with each other by synchronizing to a common server account.*” *Id.*, 2:63-67. As discussed above for ’823 claim [1b], a POSA thus would have understood from Brown’s disclosures that the disclosed synchronization process applies to changes to a file that are made on either of client devices **130** or **135** associated with the same user, which will then be synchronized to the other client device through the server. A POSA would have understood that the reference to the client devices associated with a user (such as devices **130** and **135**) are fungible in the sense that files would be synchronized bi-directionally

between client devices **130** and **135** as files are modified on either of the devices.

894. A POSA would have understood and found it obvious that a second file on the second client device (for example, laptop computer **135**) would be received via the second client's application to the server application when that file is modified. Brown describes a synchronization process with respect to desktop client **130** that starts by checking whether the server has any updates for the user's account, which would be updates from other client devices of that user such as laptop client **135**. Brown, 10:18-27 ("Otherwise, server 105 has changes that client 130 lacks. Server 105 then sends the SSD to client 130 (in response to a request for the SSD by the client), informing the client of the pertinent changes, as shown in box 415. Specifically, the SSD includes the SSI, the SIDs of any directories that contain changes since the last time client 130 synchronized with server 105, the SIDs of all items (files and directories) in the changed directories, and the metadata of all items (files and directories) that have been changed since the last time client 130 synchronized with server 105."). Later, client **130** can send all pertinent changes made on client **130** to server **105** to update other client devices associated with the same user, as would be motivated by a user wanting to keep, for example, work and home computers synchronized as discussed in Brown. *Id.*, 10:53-55 ("Once client 130 has downloaded all the pertinent changes from server 105, client 130 can send

all the pertinent changes made on client 130 to server 105.”). Thus, this limitation would have been obvious for the same reasons discussed above in connection with claim [1b].

- f. [1d]: “wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and”

895. Brown in combination with Hesselink discloses wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user.

896. For example, it would have been obvious to synchronize the server with a client device, for example the laptop client 135 (“**third electronic device**”) described in Brown. The client would initiate communication with the server, and the server could respond based on a determination that the client and server are in communication. Also, it would have been obvious to use any handshake type function in TCP/IP or HTTP or any protocol, given that Brown teaches that TCP/IP and HTTP are used, or any protocol may be used: “The client SA communicates

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

with the server SFS via TCP/IP. Preferably, the client SA communicates with the server SFS using a proprietary protocol tunneled within the hypertext transport protocol (HTTP), but a person skilled in the art will recognize that other protocols can be used. ***Communication between the client SA and the server SFS is initiated by the client SA and responded to by the server SFS.***” Brown, 3:1-8.

897. Brown teaches that the synchronization is bidirectional between multiple client devices that “synchronize with each other by synchronizing to a common server account.” Brown, 2:65-67. “The client component is a client Synchronization Application (SA) that uses the server SFS to synchronize its local client data with the server. ***Client machines synchronize with each other by synchronizing to a common server account.***” *Id.*, 2:63-67.

898. A POSA would have understood and found it obvious that a **modified first electronic file copy** would have been **automatically** sent to the **third electronic device** (for example, laptop client **135**) from the server via its SFS (**first application**) **responsive to receiving the modified first electronic file copy from the second electronic device.** Brown describes a synchronization process with respect to laptop client **135** that starts by checking whether the server has any updates for the user’s account, which would be updates from other client devices of that user such as desktop client **130**. Brown, 10:18-27 (“Otherwise, server 105 has changes

that client 130 lacks. Server 105 then sends the SSD to client 130 (in response to a request for the SSD by the client), informing the client of the pertinent changes, as shown in box 415. Specifically, the SSD includes the SSI, the SIDs of any directories that contain changes since the last time client 130 synchronized with server 105, the SIDs of all items (files and directories) in the changed directories, and the metadata of all items (files and directories) that have been changed since the last time client 130 synchronized with server 105.”). Later, client **135** can send all pertinent changes made on client **135** to server **105** to update other client devices associated with the same user, as would be motivated by a user wanting to keep, for example, work and home computers synchronized as discussed in Brown. *Id.* 10:53-55 (“Once client 130 has downloaded all the pertinent changes from server 105, client 130 can send all the pertinent changes made on client 130 to server 105.”).

899. Brown teaches that the server application (SFS) automatically transfers the modified file received from the **second electronic device** via the server to the **third electronic device** to replace an older version of the file stored on the third electronic device. As discussed above with respect to ’823 claim [3c], it would have been obvious to update another device associated with the same user by transferring an updated version of a file received from one user device to the other. Synchronization in Brown’s system is bidirectional between multiple client devices

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

that “synchronize with each other by synchronizing to a common server account.”

Brown, 2:65-67. “Client machines synchronize with each other by synchronizing to a common server account.” *Id.*, 2:63-67. After the server account receives a modified file from one client device, the server application transfers that file to another client device associated with the common server account. *Id.*, 10:18-27, 10:53-55 (describing that when the “server 105 has changes that client 130 lacks,” then the client will “download[] all the pertinent changes from server 105”).

900. Brown expressly teaches that a downloaded file **replaces an older version of the file** if it exists: “On the other hand, if less than a threshold number of message digests in the MDAs match, then ***the entire file is downloaded from the server***. Once the download file is completely constructed, the client inserts the download file into its final location, ***replacing an older version of the file if it exists***.” Brown, 13:18-23.

901. This element is also obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission to the offline client. Hesselink, ¶ 183 (“As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. Computers that are offline at such a time

will be automatically updated as soon as those computers go online again.”); ¶ 197 (“Changes in any of this information that are made by the user at that user module location while that user module is unable to connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s).”).

902. In addition, a POSA would have found it obvious in view of Hesselink to use Brown’s server to automatically push updates from the server to connected clients as soon as file updates occur. For example, Hesselink discloses: “As noted above, *when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location.*” Computers that are offline at such a time will be automatically updated as soon as those computers go online again.” Hesselink, ¶ 183. A POSA would have been motivated to improve Brown’s system, to the extent it was deficient, with this improvement from Hesselink for the reasons discussed above in connection with claim 1 and, in particular, in view of Brown’s stated goal to provide “a way to

maintain distributed files across multiple clients, maintaining currency at each client as changes are made” and Brown’s teaching that “a person skilled in the art [would] recognize that *the client SA can use any scheduling algorithm to initiate synchronization*,” including Hesselink’s algorithm for automatic updating. Brown, 1:46-50, 3:42-46.

- g. [1e]: “**automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;**”

903. Brown in combination with Hesselink discloses automatically transferring the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user.

904. For the reasons discussed above, Brown’s “clients” are fungible. Thus, for the reasons discussed above in connection with claim [1d], it would have been obvious to automatically transfer the modified “**second electronic file copy**” from Brown’s server, via the server’s “**first application**,” to the “**second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having**

the content modified by the user.”

- h. [1f]: “wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.”**

905. Brown in combination with Hesselink discloses wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.

906. Brown discloses that client files reside on client devices as part of the operating system file systems on the client devices, such as Windows, in which users may perform save operations on files. Brown, 7:3-20. “Data files reside within the directory on the client as files native to the operating system of the client machine. Like the server metadata, the client metadata includes file and directory items with fields such as name, update and create time, and file length.” Brown, 6:64-7:1. “In an embodiment of the invention running under the Windows XP/2000/NT operating system, the client SA monitors file system activity using a filter driver. In another embodiment of the invention running under the Windows 9x operating systems, the client SA monitors file system activity using a VxD.” *Id.*, 7:6-11. Brown further discloses: “The client SA monitors file system activity within the user’s directory on

the client.” *Id.*, 7:3-4. A POSA would have understood the aforementioned “file system activity” to necessarily include “save operation[s]” on files, and it would have been obvious in view of these disclosures to consider references like Hesselink that further elaborate on “**automatically transfer[ring] the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.**”

907. Hesselink discloses and renders obvious that the modified file is uploaded from the client machine upon determining that a save operation has been performed on it. A POSA would have understood the save operation to be an obvious triggering event for the file transfer and, thus, would have found it obvious to incorporate this functionality into Brown’s systems and methods: “As the user works on the file, such as by adding to the file, modifying or editing, ***each time that the user saves his/her work***, the system compares the differences between the saved version of the file and the previous version of the file, and, as long as the user module remains connected to the network, ***the changes/differences are transmitted over the network*** to the remote device module, which are saved there, thereby maintaining the home file (i.e., official reference file, source file, master file or DataClient file) up to date with the latest revisions, as described above.” Hesselink, ¶ 189. “Alternatively these processes may be set as to selective computers, which may be

any subset of the full set of computers that are connected and subscribed. Thus, in the example provided above with allowing access of a user's pictures by his parents, the user may set the system so that ***when the user saves pictures to his local computer***, in a file that has been granted permission to his parents, ***the system simultaneously also saves*** the pictures to the parents' hard drive (***remote device***).”
Id., ¶ 214.

2. Claim 3

908. Claim 3 recites: “**The system of claim 1 wherein the first application is configured to store the modified first electronic file copy to a memory device associated with the first electronic device.**” It would have been obvious to practice this claim in view of Brown.

909. Brown discloses that the server SFS stores the modified file in memory. “FIG. 1 shows a server with several clients accessing files stored on a server SFS on the server, according to an embodiment of the invention. In FIG. 1, server **105** includes server SFS **110**. Server **105** includes all the typical elements of a server, such as a central processor, **memory**, bus, disk space, etc. Typically, server SFS **110** is installed on a hard disk drive within server **105**, but a person skilled in the art will recognize other forms of media usable by server **105**: for example, removable media or optical media. **Stored on server SFS 110** are folders **115**—

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

1, 115–2, and 115–3. Although FIG. 1 shows only three such folders, a person skilled in the art will recognize that there can be more or fewer folders. Each of the folders is assigned to a user. Once the user has logged in, the user can access the files and directories (cumulatively called directory entries) within the folder. For example, folder **115–1** is shown with three files **117–1, 117–2, and 117–3.**” Brown, 3:54-4:3. “For file uploads, a file is first uploaded to a temporary file on the server. Then, *after the file data is on the server, the server SFS inserts the file and its metadata into the user’s server account database* in an atomic operation.” *Id.*, 4:64-5:1. “Once the file has been completely uploaded, the server then stores the file in the specified directory and updates the file metadata.” *Id.*, 13:1-3.

3. Claim 4

910. Claim 4 recites: “**The system of claim 3, wherein the first application is configured to store the modified first electronic file copy to the memory device associated with the first electronic device when the third electronic device is not in communication with the first electronic device.**” It would have been obvious to practice this claim in view of Brown in combination with Hesselink.

911. Brown’s server SFS stores the modified file when the client device is not in communication. Brown teaches that there are times when the client devices are “disconnected from the server.” Brown, 17:64-66. Only when the client device

synchronizes with the server, then the client device downloads the file from the server, as Brown describes in describing the synchronization process: “The filter drivers can track changes made locally, ***even while disconnected from the server.*** The user can ***later log in to the server*** and be authenticated, at which point changes can be migrated to the server.” *Id.*, 17:64-18:1.

912. This element is also obvious in view of Hesselink, which discloses determining whether a client is offline and, if so, storing a file for later transmission to the offline client. Hesselink, ¶ 183 (“As noted above, when data is updated, such as when a file is edited, for example, the system may be configured to automatically update all or some designated subset of computers on the network as soon as the update has been performed at one location. ***Computers that are offline at such a time will be automatically updated as soon as those computers go online again.***”); ¶ 197 (“The present system is persistent and therefore supports user modules that may not always be able to communicate (i.e., over the network) with the device module. Each user module both file data and management information such as usernames, authentications, device information, permissions, other subscription information, and other information that may be used in managing communication with remote devices, in cache memory. Changes in any of this information that are made by the user at that user module location while that user module is unable to

connect with the network are stored locally in cache. Changes that can be stored include all normal drive operations, thus the interface can support normal functionality. Once the user module is able to re-establish a connection to the network, updates (including any updates to file data as well as updates to management information) may be automatically transmitted from that particular user module to the appropriate device module(s). In turn, the updated device module(s) then update any other user modules that are affected by the changes. Hence when the user works locally on a file, the user does not need to be concerned with managing the transmission or other movement of files back and forth between the user module (local computer) and the device module (remote storage device), as the system manages this automatically for the user.”). It would have been obvious to incorporate this sort of functionality into Brown in view of Brown’s aim to provide “*a way to maintain distributed files across multiple clients, maintaining currency at each client as changes are made.*” Brown, 1:46-50. See, e.g., Hesselink, ¶ 172 (“NAS interface 148 operably connects one or more storage device(s) 154 to connection server 14, thereby *creating central storage for users of the system*. Such central storage may be extremely useful for backups or other storage.”).

4. Independent Claim 8

913. Claim 8, for purposes of my analysis, is substantially similar to claim 1

Declaration of Ravin Balakrishnan, Ph.D. in Support of
 Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
 10,289,607, 10,642,787, 10,754,823, and 11,003,622

except for the preamble, which recites: “A method implementable in an electronic system having a storage component, the electronic system being coupled to a first electronic device having stored thereon a first electronic file and a second electronic device having stored thereon a second electronic file, the first electronic device and the second electronic device each being associated with a user.” As the following table shows, both claims otherwise recite substantively identical claim limitations:

Claim 1 (system)	Claim 8 (method)
A system comprising:	A method implementable in an electronic system having a storage component, the electronic system being coupled to a first electronic device having stored thereon a first electronic file and a second electronic device having stored thereon a second electronic file, the first electronic device and the second electronic device each being associated with a user, the method comprising:
a first electronic device configured to selectively execute a first application, the first electronic device being in communication with a second electronic device and a third electronic device, each associated with a user wherein the first electronic device is configured to:	
receive from a second application executable on the second electronic device a copy of a first electronic file automatically transferred from the second application when the user modifies a content of the first electronic file; and	receiving from the first electronic device a copy of a first electronic file modified by the user, the copy of the modified first electronic file being automatically provided to the electronic system by the first electronic device when a content of the first electronic file is modified by the user;
wherein the first electronic device is further configured to receive from a third application executable on the third electronic device a copy of a second electronic file automatically transferred	receiving from the second electronic device a copy of a second electronic file modified by the user, the copy of the modified second electronic file being automatically provided to the electronic

from the third application when the user modifies a content of the second electronic file; and	system by the second electronic device when a content of the second electronic file is modified by the user;
wherein the first application is further configured to automatically transfer the modified first electronic file copy to the third electronic device to replace an older version of the first electronic file stored on the third electronic device with the modified first electronic file copy having the content modified by the user; and	automatically transferring the modified first electronic file copy to the second electronic device to replace an older version of the first electronic file stored on the second electronic device with the modified first electronic file copy having the content modified by the user; and
automatically transfer the modified second electronic file copy to the second electronic device to replace an older version of the second electronic file stored on the second electronic device with the modified second electronic file copy having the content modified by the user;	automatically transferring the modified second electronic file copy to the first electronic device to replace an older version of the second electronic file stored on the first electronic device with the modified second electronic file copy having the content modified by the user;
wherein the second application automatically transfers the copy of the modified first electronic file to the first electronic device upon determining that a save operation has been performed on the modified first electronic file.	wherein the first electronic device automatically transfers the copy of the modified first electronic file to the electronic system upon determining that a save operation has been performed on the modified first electronic file.

914. For the reasons discussed above in connection with claim 1, and the reasons below, claim 8 would have been obvious.

915. Brown's server includes disk space and one or more processors programmed with computer program instructions that may be executed to cause operations. Brown, 3:57-59 ("Server 105 includes all the typical elements of a

server, such as a central **processor**, memory, bus, **disk space**, etc.”). As discussed above in connection with ’823 element [1a], Brown and Hesselink both disclose servers containing hard disk drives, which could be used to implement the “**storage component**” in this claim. As discussed above in connection with claim 1, the client devices described in Brown may store files and may be associated with the same user. *See, e.g.*, Brown, 15:22-25 (“At step 1065, whether the client downloaded the entire file or only the changed blocks, the client moves the downloaded file to the directory in which it is to be stored.”).

5. Claim 10

916. This claim would have been obvious for the reasons below and those discussed in connection with claims 3 and 8. Claim 10 recites: “The method of claim 8, wherein the electronic system is configured to store the modified first electronic file copy to the storage component.” As discussed above in connection with claim 8, Brown’s server includes a storage component, and the server is configured to store the modified first electronic file copy to the storage component for the reasons discussed above in connection with claims 3 and 8.

6. Claim 11

917. This claim would have been obvious for the reasons below and those discussed in connection with claims 4, 8, and 10. Claim 11 recites: “The method of

claim 10 wherein the electronic system is configured to store the modified first electronic file copy to the storage component when the second electronic device is not in communication with the electronic system.” As discussed above in connection with claim 8, Brown’s server includes a storage component, and the server is configured to store the modified first electronic file copy to the storage component when the second electronic device is not in communication with the electronic system for the reasons discussed above in connection with claims 4, 8, and 10.

VIII. NO SECONDARY CONSIDERATIONS OF NON-OBVIOUSNESS

918. As explained in Part III.B, I understand that “secondary considerations” may be used as indicia of non-obviousness, so long as any such consideration has a nexus to the claimed invention. I am not aware of any information (such as commercial success, unexplained results, long felt but unsolved need, industry acclaim, simultaneous invention, copying by others, skepticism by experts in the field, and failure of others) suggesting that the claims addressed in this Declaration are not obvious. If Patent Owner later provides information it claims relates to secondary considerations, I reserve the right to respond.

IX. ENABLEMENT AND REASONABLE EXPECTATION OF SUCCESS

919. I understand from Petitioner’s counsel that in an *inter partes* review,

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

the petitioning party does not have a burden to show that the prior art is enabling. In any event, in my opinion, the prior art references that I have analyzed in this Declaration provide sufficient detail to enable a person of ordinary skill in the art to practice the limitations of the claims addressed in this Declaration without undue experimentation. Based on this detail, in my opinion, a person of ordinary skill in the art would have had a reasonable expectation that the prior art references could have been successfully combined in the manner that I describe throughout this Declaration.

920. As discussed above, all of the asserted references are analogous and complementary, disclosing software or software components/utilities that were designed to facilitate file synchronization across multiple client devices and related user interface functionality. The Hesslink (¶¶ 128-129, 133, 187, 203, 204, 220), Shappell (cover page at [73])), Rao (cover page at [73])), and Moromisato (cover page at [73])) references describe the workings of commercially developed Microsoft technology. As such a POSA would have had every reason to reasonably expect success in combining the abovementioned features and functionality from these references—developed in connection with successful companies in the space for file synchronization and user interface technology at the time—with Sigurdsson and Brown.

921. In addition, I understand that a patent need not teach, and preferably omits, what is well known in the art. Therefore, the relatively brief treatment that the prior art references give to some of the underlying technologies and techniques makes clear that they were well known and well within the capabilities of a person of ordinary skill in the art.

922. A person of ordinary skill in the art, therefore, would not have required disclosures any more detailed than the disclosures in the prior art to successfully combine and apply their teachings in the manner I have detailed in this Declaration.

X. CONCLUSION

923. I reserve the right to offer additional opinions relevant to the validity of the Challenged Patents and Claims at issue and/or offer testimony in support of the Declaration.

924. In signing this Declaration, I recognize that the Declaration will be filed as evidence in a contested case before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case. If required, I will appear for cross-examination at the appropriate time.

925. I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be

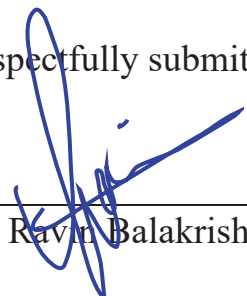
Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

true, and further that these statements were made with the knowledge that willful
false statements and the like so made are punishable by fine or imprisonment, or
both, under 18 U.S.C. § 1001.

Declaration of Ravin Balakrishnan, Ph.D. in Support of
Petitions for *Inter Partes* Review of U.S. Patent Nos. 9,143,561, 10,067,942,
10,289,607, 10,642,787, 10,754,823, and 11,003,622

Dated: January 3, 2023

Respectfully submitted,



Dr. Ravin Balakrishnan

EXHIBIT A

Ravin Balakrishnan

Curriculum Vitae

Contact information

Address: Department of Computer Science
University of Toronto.
10 King's College Road
Toronto, Ontario
Canada M5S 3G4

voice: (416) 835-1600
email: ravin@dgp.toronto.edu
web: www.dgp.toronto.edu/~ravin

Education

Ph.D. Computer Science, University of Toronto, Canada. Feb 2001.

Thesis: "Issues in Bimanual Interaction for Computer Graphics."

Advisor: Prof. Bill Buxton

Advisory Committee: Prof. Mark Chignell, Prof. Eugene Fiume, Prof. Scott MacKenzie

M.Sc. Computer Science, University of Toronto, Canada. Jan 1997.

Thesis: "The Evolution and Evaluation of a 3D Input Device."

Advisor: Prof. Bill Buxton

B.Sc. (1st Class Honours), Computer Science, University of New Brunswick, Canada. May 1993.

Employment

Chair, Department of Computer Science, University of Toronto, 2015 – 2019.

Professor (tenured), Department of Computer Science, University of Toronto, 2011 – present.

Canada Research Chair (CRC-Tier II) in Human-Centred Interfaces, Department of Computer Science,
University of Toronto, March 2006 – 2016.

Associate Professor (tenured), Department of Computer Science, University of Toronto, 2006- 2011.

Canada Research Chair (CRC-Tier II) in Human-Centred Interfaces, Department of Computer Science,
University of Toronto, March 2006 – 2016.

Associate Chair, Research and Industrial Relations, Department of Computer Science, University of Toronto. July 1, 2008 – June 30, 2010.

Assistant Professor (tenure-track), Department of Computer Science, University of Toronto, 2001 –2006. Appointed as Full Member of the Graduate Faculty in 2001.

Researcher, User Interface Research Group, Alias|wavefront (now Autodesk), Aug 1996 – July 2001.

Conducted basic and applied research on interfaces for interactive 3D graphics, alternate multi degree-of-freedom input devices, two-handed input, haptic feedback interfaces, tablet based input, and large scale displays. Some of the ideas and interaction techniques developed in this research have been incorporated into Alias|wavefront's market leading software products for 3D design, modeling, and animation.

Instructor, Dept. of Computer Science, University of Toronto, May 1995 – Aug 2000, on a per course basis.

Lectured and took complete responsibility, including managing several teaching assistants, for various undergraduate courses (listed in section entitled "Teaching" below).

Teaching Assistant, Dept. of Computer Science, University of Toronto. Sep 1993 - Dec 1996.

Conducted tutorials and marked assignments, tests, and exams for various courses (listed in section entitled "Teaching" below).

Research Assistant, Human Computer Interaction Laboratory, University of New Brunswick, May - Sept 1993.
Worked with Prof. Colin Ware on empirical research into the effects of lag in fishtank virtual reality environments. Also helped design, build, and evaluate a haptic feedback input/output device.

Programmer Analyst, Indigo Software Ltd, Ottawa, Canada. Sep - Dec 1990, May - Aug 1991, Jan - May 1992.

Visiting and consulting activities

Visiting researcher, HPLabs Bangalore, India. Feb 2009.

Visiting researcher, Microsoft Research - Redmond, Beijing, Bangalore and Cambridge labs. July 2007 – June 2008.

Visiting professor, Laboratoire de Recherche en Informatique (LRI), Université Paris-Sud. France. June-July, 2006

Visiting researcher, Mitsubishi Electric Research Lab (MERL), Cambridge, MA, USA. Feb 2005 – Mar 2007.

Expert witness for Broadband iTV Inc. through Kramer Day. Defendant: Comcast, Charter, Altice. ITC Investigation No 337-3616. 2022 – ongoing.

Expert witness for MyPort Inc. through One LLP. Defendant: Samsung. Case No. 2:22-cv-00114 in the US District Court for the Eastern District of Texas Marshall Division. 2022 – ongoing.

Expert witness for Blue Yonder Group Inc. through Cravath. Defendant: Kinaxis. US District Court for the Northern District of Texas. Civil Action No. 3:20-cv-3636. 2021 – ongoing.

Expert witness for G Holdings through Hall Booth Smith. Defendant: Samsung. US District Court for the Eastern District of Texas Marshall Division. Civil Action No. 2:20-cv-342. 2021. Prepared one declaration. Case settled.

Expert witness for MLB Advanced Media through Goldman Ismail LLP. Plaintiff: Sportvision, Inc. Case No. 18-cv-03025, in the United States District Court for the Southern District of New York. 2021 – ongoing. Prepared one report.

Expert witness for Roku through Dickinson-Wright in an IPR Case No. IPR2022-00818. 2022. Prepared one report.

Expert witness for Roku through Dickinson-Wright. Defendant: UEI. ITC Investigation No. 337-TA-1264. 2021 – 2022. Prepared several reports. One deposition, testified at trial.

Expert witness for Roku through Dickinson-Wright. Plaintiff: UEI. ITC Investigation No. 337-TA-1200. 2020 – 2021. Prepared several reports. One deposition, testified at trial.

Expert witness for Dell through Alston Bird. Plaintiff: Neodron. 2020. Case No. 1-cv-003182020. Prepared one report. Case settled.

Expert witness for Content Square SAS v. Medallia Inc. through Cooley then Troutman Pepper. IPR IPR2022-00152. 2021. Prepared on report.

Expert witness for Apple through Morrison Foerster. Plaintiff: MemoryWeb LLC. Case No. 6:21-cv-521-ADA. 2021-2022. Did some preliminary work. Case settled.

Expert witness for Spacetime3D through Sussman Godfrey. Defendant: Samsung. Case 2:19-cv-00372-JRG in the U.S. District Court for the Eastern District of Texas. 2020 – 2021. Prepared several reports, one deposition. Case settled.

Expert witness for Supercell Oy through Fenwick and West. Defendant: Gree. IPR Proceeding. 2020. Prepared several reports.

Expert consultant for Abbott Labs UK through Taylor Wessing LLP. 2020-2021.

Expert witness for Kannuu through Progress LLP in an IPR proceedings (IPR2020-00736, -00737, -00738, -00739, -00740). Plaintiff: Samsung. 2020 – 2021. Prepared several reports. One deposition.

Expert witness for Microsoft through Sidley Austin LLP. Plaintiff: Zeroclick. 2020. Case settled. Retained, but no work done.

Expert witness for Sony through Kirkland and Ellis LLP. Plaintiff: Bot M8. Case No. 3:19-cv-07027-WHA, In the US District Court in the Northern District of California. 2020 – ongoing. No work done to date.

Expert witness for Facebook through Cooley LLP. Plaintiff: Mirror Worlds Technologies. Case No. 1:17-cv-03473-JGK in the U.S. District Court in the Southern District of New York. 2020 – ongoing. Prepared a report. One deposition.

Expert witness for Facebook through Cooley LLP. Plaintiff: Blackberry. Case No. 18-cv-1844 (C.D. Cal.). Prepared several reports, one deposition. Case stayed in 2020 pending IPRs.

Expert witness for Bumble through Cooley LLP. Plaintiff: KinectUs LLC. Case No: 6-20-cv-00942-ADA, Western District of Texas. 2021. One report. Case settled.

Expert witness for Bumble through Cooley LLP. Plaintiff: Match. Case No: 6:18-cv-00080-ADA, Western District of Texas. 2020. One report and one deposition. Case settled.

Expert witness for Polaris PowerLED Technologies Ltd., through Feinberg Day. Defendants: LG. In the US District Court for the Central District of California, Civil Action No: 8-18-cv-0125-JVS-DFM. 2020. No work done to date.

Expert witness for Polaris PowerLED Technologies Ltd., through Feinberg Day. Defendants: Vizio. In the US District Court for the Central District of California, Civil Action No: 8-18-cv-01571-JVS-DFM. 2019 – 2020. Prepared one report.

Expert witness for NexStep Inc. through Kramer Levin LLP. Defendant: Comcast. Case C.A. No. 19-1031 (RGA)(SRF) in the U.S. District Courts for the State of Delaware. Judge: Hon. Richard Andrews. 2019 – 2021. Prepared one report, one deposition, testified at trial.

Expert witness for iTools Media Inc., through Gilberts LLP. Plaintiffs: Caps Group Inc., and CASLCure Inc. Court File No T-1543-17. 2019. Prepared one report. Case settled.

Expert witness for Polaris PowerLED Technologies Ltd., through Dentons and Feinberg Day. Defendants: Samsung. No. 2-17-cv-00715-JRG, in the U.S. District Court for the Eastern District of Texas. 2018 – 2019. Prepared several reports. One deposition. Case settled.

Expert witness for Charter., through Quinn Emanuel. Plaintiff: Sprint Communications. 2018. Case settled. No work done.

Expert witness for Immersion Corp, through Morrison and Foerster. Defendants: Motorola Mobility and Samsung. Case No. 1:17-cv-01081-RGA District of Delaware, Civil Action No 2:17-cv-572-JRG, Eastern District of Texas. 2017 – 2019. Prepared several reports, several depositions. Cases settled in 2019.

Expert witness for Immersion Corp. through McKool Smith. IPR2018-00588. 2018. Prepared one report.

Expert witness for 3Shape Inc. through Buchanan Ingersoll & Rooney. Plaintiff: Align Technology, Inc. IPR proceedings IPR2018-00197, IPR2018-00198. Prepared one report. One deposition. 2018 – 2019.

Expert witness for Rovi Corp, through Gilberts LLP then Lencner Slaght. Plaintiffs, Videotron, Bell, Telus. 2017 – 2020. Prepared reports. Testified at Trial. Federal Court of Canada.

Expert witness for Phillips, through Fitzpatrick, Cella, Harper and Scinto. Plaintiff: HTC corp. IPR proceedings IPR2017-00857. Prepared report. 2017 – 2018.

Expert witness for Rovi Corp, through Ropes and Gray. Plaintiff: Comcast Corp. IPR proceedings for several patents. 2017 – 2018. Prepared several reports, two depositions.

Expert witness for Rovi Guides Inc. through McKool Smith. Plaintiff: Comcast Corp. in Rovi Guides Inc., et al. v. Comcast Corp., et al., No. 2:16-cv-321 (EDTX); Rovi Guides Inc. v. Comcast Corp., et al., No. 2:16-cv-322 (EDTX); Certain Digital Video Receivers and Hardware and Software Components Thereof, Inv. No. 337-TA-1001, TA-3135 (ITC); Comcast Corp., et al. v. Rovi Corp., et al., No. 1:16-cv-03852 (SDNY). 2016 – 2020. Testified at two ITC hearings.

Expert witness for Oculus and Palmer Luckey through Cooley, LLP, at the US District Court for the Northern District of California. Plaintiff: Total Recall Technologies. Case No. 3:2015cv02281. 2016 – 2021. Judge: Hon. William Alsup. Prepared one report. One deposition. Attended trial.

Expert witness for Nuance Communications through Weil Gotshal and Manges LLP, in an IPR proceeding at the US Patent Office. 2016. Draft report prepared. Case terminated.

Expert witness for Oculus, Facebook and Palmer Luckey through Cooley, LLP, in a trade secret action at the US District Court for the Northern District of Texas. Plaintiff: ZeniMax Media Inc. and ID Software LLC. Case No. 3:14-cv-01849-P. 2015-2016. Several reports and depositions. Testified at trial.

Expert consultant for ZTE through Upshaw PLLC. 2015. Did some preliminary prior art research.

Expert witness for Apple through O'Melveny & Myers, LLP, in a patent infringement action at the US District Court for the District of Northern California. Plaintiff: OpenTV, Inc. and NagraVision, SA. Case No: 14-CV-1622. Case settled. Retained for the case but no work done.

Expert witness for Mojo Media Inc. through Morrison Foerster LLP, in a patent interference action. Interference No. 106,005, Mojo Media, Inc. v. Snapchat, Inc. 2014 – 2015. Prepared several reports. Case terminated in 2015.

Expert witness for Queens University and PARTEQ Research and Development Innovations (plaintiff) through Susman Godfrey LLP, and Sterne, Kessler, Goldstein & Fox in a patent infringement action at the US District Court for the Eastern District of Texas and an IPR proceedings at the US Patent office. Case: 2:14-cv-53. Defendant: Samsung. 2014-2016. Prepared several reports, two depositions.

Expert witness for Greenwood Publishing Group, Inc. d/b/a Heinemann (defendant) through Wilmer Hale in a patent infringement action at the US District Court for the Southern District of New York. Case: 1:13-cv-02687-LTS-RLE. Plaintiff: Amplify Education Inc. 2014. Prepared one declaration on claim construction. Case settled in 2014.

Expert witness for Microsoft (defendant) through Sidley Austin LLP, in a patent infringement action at the US District Court for the District of Maine. Case: 2:12-cv-333-DBH. Plaintiff: SurfCast Inc. 2013 – 2014. Prepared several reports, one deposition.

Expert witness for VMware (defendant) through Morrison Foerster, LLP, in a patent infringement action at the US District Court for the District of Delaware. Case: 13-cv-116. Plaintiff: UbiComm LLC. 2013. Prepared several declarations.

Expert witness for Hockeyline Inc. (plaintiff) through Freitas Tseng & Kaufmann LLP, in a patent infringement action at the US District Court for the Southern District of New York. Case: 1:13-cv-01446-CM. Defendant: STATS LLC. 2013 – 2015. Prepared one report. Case terminated in 2015.

Expert witness for Pixcir Microelectronics (defendant) through Nixon Peabody LLP, in a patent infringement action at the US District Court for the District of Nevada. Case: 2:10-cv-00014-GMN-PAL. Plaintiff: Elan Microelectronics. 2013. Prepared one report. Case settled in 2013.

Expert witness for Nuance (complainant) through Weil, Gotshal & Manges, LLP, in a patent infringement action at the ITC. Investigation No. 337-TA-785. Respondent: Cootek. Several reports and one deposition. Case settled in 2013.

Expert witness for Yahoo (defendant) through Morrison Foerster, LLP, in a patent infringement action. Plaintiff: Interval. 2012. Retained, but no work done.

Expert witness for Immersion (complainant) through Morrison Foerster, LLP, in a patent infringement action at the ITC, Investigation No: 337-TA-834 and in the District Court for the District of Delaware. Respondent: HTC. 2012-2015. Several reports, two depositions. ITC case terminated in 2012, District court case settled in 2015.

Expert witness for Apple (plaintiff) through Freehills in a patent infringement action in Australia. Defendant: Samsung. 2012-2013. Prepared several reports and testified at trial.

Expert witness for Apple (plaintiff) through Freshfields Bruckhaus Deringer in a patent infringement action in the UK. Defendant: HTC. 2012. Several reports. Case settled in Nov 2012.

Expert witness for Apple (plaintiff) through Freshfields Bruckhaus Deringer in a patent infringement action in the Netherlands. Defendant: Samsung. Prepared one declaration and attended injunction hearing.

Expert witness for Apple (plaintiff) through Gibson Dunn, LLP, in a patent infringement action at the US District Court for the Northern District of California. Defendant: Samsung. Case No. 12-cv-00630-LHK. 2012. Judge: Hon. Lucy Koh. One report, one deposition for preliminary injunction.

Expert witness for Apple (complainant) through Feinberg Day, LLP, in a patent infringement action at the ITC. Respondent: HTC. Investigation No. 337-TA- 797. 2011-2012. ALJ: Hon. E. James Gildea. Several expert reports, one deposition, testified at trial.

Expert witness for Apple (defendant) through O'Melveny & Myers, LLP, in a patent infringement action at the US District Court for the District of Delaware. Plaintiff: MobileMedia Ideas LLC. Case No: 10-258-SLR. 2011-2016. Judge: Hon. Sue Robinson. Several reports, several depositions, testified at two trials.

Expert witness for Apple (respondent) through Wilmer Hale, LLP, in a patent infringement action at the ITC. Complainant: Samsung. Investigation No. 337-TA- 794. 2011-2012. ALJ: Hon. E. James Gildea. Several expert reports, one deposition, testified at trial.

Expert consultant for Apple through Wilmer Hale LLP. Plaintiff: Nokia. Case No 09-791-GMS in the US District Court for the District of Delaware. 2011. Did some preliminary work.

Expert witness for Apple (plaintiff) through Morrison & Foerster, LLP, in a patent infringement action at the US District Court for the Northern District of California. Defendant: Samsung. Case No. 11-cv-01846 LHK. 2011-2013. Judge: Hon. Lucy Koh. Several reports, several depositions, testified at technology tutorial and three trials.

Expert witness for Apple (complainant) through Morrison & Foerster, LLP, in a patent infringement action at the ITC. Respondent: Samsung. Investigation No. 337-TA- 796. 2011-2012. ALJ: Hon. Thomas Pender. Several reports, one deposition, testified at technology tutorial and at trial.

Expert witness for Apple (complainant) through Weil, Gotshal & Manges, LLP, in a patent infringement action at the ITC. Respondent: Motorola. Investigation No. 337-TA- 750. 2011. ALJ: Hon. Theodore Essex. Several reports, one deposition, testified at trial.

Expert witness for Apple (defendant) through Weil, Gotshal & Manges, LLP, in a patent infringement action in the US District Court for the Northern District of California. Plaintiff: Elan Microelectronics. Case No. 5:09-cv-01531-RS (N.D. Cal) 2010-2012. Judge: Hon. Richard Seeborg. Several declarations. Case settled in 2012.

Expert witness for Apple (respondent) through Weil, Gotshal & Manges, LLP, in a patent infringement action at the ITC. Complainant: Elan Microelectronics. Investigation No. 337-TA-714. 2010-2011. CALJ: Hon. Paul Luckern. Several reports, two depositions, testified at Markman hearing and at trial.

Expert witness for Nintendo (respondent) through Covington and Burling, LLP, in a patent infringement action at the ITC. Complainant: Hillcrest Laboratories. Investigation No. 337-TA-658. 2008-2009. CALJ: Hon. Paul Luckern. Several reports, two depositions, testified at trial.

Entrepreneurial Activities

Co-founder, Conceptualiz www.conceptualiz.com. Developing surgical planning technology.

Co-founder, Arcestra www.arcestra.com (formerly Sketch2 Corp). Developing software for designing 3D building interior layouts. The concept grew out of research conducted in my lab with my colleague Prof. Karan Singh and graduate students Steven Tsang and Abhishek Ranjan.

Co-founder, Bump Technologies Inc. www.bumptop.com. Acquired by Google in May, 2010. Developed a physically realistic desktop user interface system, which grew out of research conducted in my lab with graduate student Anand Agarawala.

Research endeavours

Human-computer interaction, computer mediated human-human interaction, data visualization, information and communications technology for development.

Specific research interests:

Input devices, sensing technologies, interaction techniques. In particular, gestural, sketching, and multi degree-of-freedom interaction, interfaces to small and/or mobile computers, interfaces to displays of the future.

User interfaces and interaction techniques for data visualization.

Empirical evaluation of user interfaces including associated metrics and predictive models of human performance.

Ambient, pervasive, and ubiquitous computing.

Computer supported collaborative work (CSCW).

Interfaces and systems for educational and healthcare applications in developing economies.

Awards, fellowships, scholarships

Canadian Human Computer Communications Society Achievement Award, 2020

Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2016.

10-year impact award at the ACM Interactive Tabletops and Surfaces conference 2016 for paper published in 2006.

Two best paper honorable mention awards at the ACM CHI 2016 Conference on Human Factors in Computing Systems.

Best paper honorable mention award at the ACM CHI 2013 Conference on Human Factors in Computing Systems.

Elected to the ACM CHI Academy in 2011.

Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2010.

Best paper award at the ACM CHI 2010 Conference on Human Factors in Computing Systems.

Best paper honorable mention award at the ACM CSCW 2010 Conference on Computer Supported Cooperative Work.

Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2009.

Best paper honorable mention award at the ACM CHI 2009 Conference on Human Factors in Computing Systems.

Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2008.

Best paper honorable mention award at the ACM CHI 2008 Conference on Human Factors in Computing Systems.
 Alfred P. Sloan Research Fellowship. 2007-2009. US\$45,000.
 Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2007.
 Canada Research Chair (CRC-Tier II) in Human-Centred Interfaces, University of Toronto, March 2006 – 2016.
 Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2006.
 Best paper honorable mention award at the ACM CSCW 2006 Conference on Computer Supported Cooperative Work.
 Best student paper award at the ACM UIST 2006 Symposium on User Interface Software and Technology.
 Best paper award at the Graphics Interface 2005 Conference.
 Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2005.
 Best paper award at the ACM CHI 2005 Conference on Human Factors in Computing Systems.
 Best paper award at the ACM UIST 2004 Symposium on User Interface Software and Technology.
 Dean's Excellence Award, Faculty of Arts and Science, University of Toronto. 2004.
 Ontario Premier's Research Excellence Award (PREA). 2003. \$100,000.
 IBM Toronto Centre for Advanced Studies Faculty Fellow (2003-present).
 Bell University Laboratories Associate Chair in Human Computer Interaction, U. of Toronto, Jan 2002 – 2006.
 Ph.D. fellowship from Alias|wavefront, \$30,000 per annum plus travel/research expenses, Sep 1996 – July 2001.
 University of Toronto, Dept. of Computer Science award for outstanding Teaching Assistant, 1994.
 University of Toronto Open Fellowship, \$4500, Sep - Dec 1994 .
 University of Toronto Connaught Scholarship, \$13,500 plus tuition fees, Sep 1993 - Aug 1994.
 Harry Levine Award for outstanding graduating student in Computer Science, University of New Brunswick, 1993.
 University of New Brunswick, Dept. of Computer Science Dean's List, 1990, 1991, 1992, 1993.

Research Grants

(Unless otherwise indicated, all figures are in Canadian dollars and I am the sole investigator/recipient of the following grants and awards)

“User interfaces and visualization for variable format displays.”

NSERC discovery grant, 1 April 2012 – 31 March 2017, \$52,000 per year.

Network Investigator in the “GRAND Networks Centres of Excellence”. PI: Kellogg Booth. 2010-2013.

HPLabs India unrestricted gift. Jan 2010. US\$15,000 cash.

“Going beyond the surface”.

Microsoft Research unrestricted gift. 1 July 2009. US\$100,000 cash + hardware donation valued at approximately US\$15,000.

“3-D visualization and gestural interaction with multimodal neurological data.”

NSERC Strategic Project Grants Supplemental Competition, 1 April 2008 – 31 March 2010, \$196,000 over 2 years (\$98,000 per year).

PI: Jeremy Cooperstock. Co-investigators: Ravin Balakrishnan, Tal Arbel, Louis Collins.

“Multi channel and multi degree-of-freedom user interfaces for heterogeneous display environments.”

NSERC discovery accelerator supplement grant, 1 April 2007 – 31 March 2010, \$120,000 over 3 years (\$40,000 per year)

“Multi channel and multi degree-of-freedom user interfaces for heterogeneous display environments.”

NSERC discovery grant, 1 April 2007 – 31 March 2012, \$166,000 over 5 years (\$33,200 per year)

“Multi-Scale Environments”

INRIA "Equipes Associées" grant. 60,000 Euros over 3 years beginning Jan 2007.

PIs: Wendy Mackay and Ravin Balakrishnan

“3D Sketch Graphics Software Development Project”.

NSERC Idea to Innovation (I2I) Phase 2 grant. \$125,000 for one year, 2006.

PI: Karan Singh. Co-investigator: Ravin Balakrishnan

“Management, Access, and Visualization of Archived Meeting Transcripts”,

NSERC Collaborative Research and Development (CRD) grant. \$360,000 over 3 years beginning 1 Jan 2006.

PI: Charles Clarke. Co-investigators: Ravin Balakrishnan, Mark Chignell, Sharon Strauss.

“3D Sketch Graphics Software Development Project”.

NSERC Idea to Innovation (I2I) Phase 1 grant. \$125,000 for one year, 2005.

PI: Karan Singh. Co-investigator: Ravin Balakrishnan

“3D Sketching with a Suggestion Database”.

CITO Technical Readiness grant. \$25,000, for one year, 2005.

PI: Karan Singh. Co-investigator: Ravin Balakrishnan

“3D Sketching Software”

NSERC IPM grant. \$50,000, for one year, 2005.

PI: Karan Singh. Co-investigator: Ravin Balakrishnan

“Improved Medical Collaboration and Case Management: Capture, Structuring and Dissemination of Spoken Discussion”.

CITO research grant. \$280,000 over two years beginning 1 Sep 2004.

PI: Mark Chignell. Co-investigators: Ravin Balakrishnan, Charles Clarke, Sharon Strauss.

“Next Generation User Interfaces for Data Visualization”

NSERC strategic grant. \$263,800 over the period 1 Jan 2004 – 31 Dec 2006.

PI: Ravin Balakrishnan. Co-investigator: Karan Singh

“User Interfaces for Database Visualization”

CITO student internship program grant, \$15,000, 1 Jan – 31 Dec 2004.

“Network for Effective Collaboration Technologies through Advanced Research”

NSERC research network grant. \$5.5million over 5 years beginning Jan 2004.

PI: Ron Baecker. Co-investigators: Ravin Balakrishnan, Saul Greenberg, Carl Gutwin, Kellogg Booth, Sheelagh Carpendale, Kori Inkpen, Gerald Penn, Kostas Plataniotis, Elaine Toms, Roel Vertegaal.

“Memories of Synchronicity: Knowledge Management and Visualization of Interaction Transcripts in Innovative Collaboration Environments”

IBM-ARISE research grant. Sept 2003 – Aug 2006, \$450,000.

PI: Mark Chignell. Co-investigators: Ravin Balakrishnan, Charles Clarke.

“Mathematical Surface Representations for Conceptual Design”

Mathematics of Information Technology and Complex Systems (MITACS) grant.

Original award: May 2003 – Apr 2005, \$180,000.

Additional award: May 2005 – ongoing, \$180,000

PI: Karan Singh. Co-investigators: Ravin Balakrishnan, Eugene Fiume, Pierre Poulin, Michiel van de Panne, Richard Zhang.

“Tablet Interfaces”

Microsoft Research unrestricted gift. July 2003. US\$25,000 cash.

“Laboratory for human-computer interaction and computer graphics”

CFI New Opportunities Fund grant. 2003-2007. \$994,066. 40% CFI, 40% OIT, 20% from industrial partners

PI: Ravin Balakrishnan. Co-investigator: Karan Singh.

“Laboratory for human-computer interaction and computer graphics”

CFI Infrastructure Operating Fund grant. 2004-2006. \$132,000.

PI: Ravin Balakrishnan. Co-investigator: Karan Singh.

“User interfaces for 3D environments”

IBM Centre for Advanced Studies grant to support my Ph.D. student, Michael McGuffin. 1 Jan 2003 – 31 Dec 2005
\$102,000 cash + \$23,000 in-kind.

“Tilting text input for mobile devices.”

Microsoft Research unrestricted gift. 1 Jan 2003. US\$25,000 cash + hardware donation valued at approximately US\$5,000.

“User interfaces for volumetric displays.”

Connaught Fund matching grant, 1 April 2002 – 31 March 2003. \$30,000 (+\$15,000 from the department)

“Laboratory for large scale high resolution interaction, graphics and vision research.”

NSERC equipment grant, 1 April 2002 – 31 March 2003, \$96,792

PI: Ravin Balakrishnan. Co-investigators: Karan Singh, Kyros Kutulakos, Eugene Fiume

“Multi-channel and multi degree-of-freedom interaction with large displays.”

NSERC discovery grant, 1 April 2002 – 31 March 2007, \$140,000 over 5 years (\$28,000 per year)

“Techniques for multi-channel and multi degree-of-freedom interaction.”

Connaught startup grant, 1 July 2001 – 30 June 2003, \$10,000.

Bell University Laboratories startup funding, 1 July 2001 – 31 Dec 2003, \$60,000

Publications

Copies of the following papers and related videos are available online at www.dgp.toronto.edu/~ravin/#publications

A note on publication venues: in my primary area of research in Human Computer Interaction (HCI), the ACM Conference on Human Factors in Computing Systems (CHI) is considered one of the best forums for dissemination of research results and covers the broad spectrum of research in HCI. The ACM Symposium on User Interface Software and Technology (UIST), the ACM Conference on Computer Supported Cooperative Work (CSCW), and the European Conference on Computer Supported Cooperative Work (ECSCW) are also of similar quality, but are focused on particular sub-areas of the field. Papers in these conferences are refereed as full papers, and have an acceptance rate of around 15-25% each year. The top journals in the HCI field are the ACM Transactions on Computer-Human Interaction, the Human Computer Interaction journal, and the International Journal of Human Computer Studies. I also do research in the fields of Information Visualization and Information and Communication Technologies for Development (ICTD) where the IEEE Visualization Conference, IEEE Information Visualization Symposium, and the IEEE/ACM International Conference on Information and Communication Technologies for Development are the top publication venues with acceptance rates of around 20-30%.

Refereed journal papers

- J1. Jian Zhao, R. William Soukoreff, Ravin Balakrishnan. (2015). Exploring and modeling unimanual object manipulation on multi-touch displays. *IJHCS: International Journal of Human-Computer Studies*, 78(0). p. 68-80.
- J2. Jian Zhao, R. William Soukoreff, Xiangshi Ren, Ravin Balakrishnan. (2014). A model of scrolling on touch-sensitive displays. *IJHCS: International Journal of Human-Computer Studies*, 72(12). p. 805-821
- J3. Jian Zhao, Christopher Collins, Fanny Chevalier, Ravin Balakrishnan. (2013). Interactive exploration of implicit and explicit relations in faceted datasets. *IEEE Transactions on Visualization and Computer Graphics (Proceedings of VAST 2013)*. p. 2080-2089.
- J4. Xiaojun Bi, Seok-Hyung Bae, Ravin Balakrishnan. (2013). WallTop: Managing overflowing windows on a large display. *Human-Computer Interaction*, 29(2). p. 153-203.

- J5. Chun Yu, Ravin Balakrishnan, Ken Hinckley, Tomer Moscovich, Yuanchun Shi. (2013). Implicit bookmarking: Improving support for revisitation in within-document reading tasks. *International Journal of Human Computer Studies*, 71(3). p. 303-320.
- J6. Jian Zhao, Fanny Chevalier, Christopher Collins, Ravin Balakrishnan. (2012). Facilitating discourse analysis with interactive visualization. *IEEE Transactions on Visualization and Computer Graphics*. 18(12). p. 2639-2648.
- J7. Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, Ravin Balakrishnan. (2011). Exploratory analysis of time-series with ChronoLenses. *IEEE Transactions on Visualization and Computer Graphics*. 17 (12). p. 2422-2431
- J8. Daniel Vogel, Ravin Balakrishnan. (2010). Direct pen interaction with a conventional graphical user interface. *Human-Computer Interaction*, 25(4). p. 324 -388.
- J9. Jeremy Birnholtz, Abhishek Ranjan, Ravin Balakrishnan. (2010). Providing dynamic visual information for collaborative tasks: Experiments with automatic camera control. *Human-Computer Interaction*, 25(3). p. 261-287.
- J10. Jennifer Keelan, Vera Pavri, Ravin Balakrishnan, Kumanan Wilson. (2010). An analysis of the Human Papilloma Virus vaccine debate on MySpace blogs. *Vaccine*, 28(6). p. 1535-1540
- J11. Gery Casiez, Daniel Vogel, Ravin Balakrishnan, Andy Cockburn. (2008). The impact of control-display gain on user performance in pointing tasks. *Human-Computer Interaction*, 23(3). p. 215-250.
- J12. Michael McGuffin, Ravin Balakrishnan. (2005). Fitts' law and expanding targets: Experimental studies and designs for user interfaces. *ACM Transactions on Computer-Human Interaction*, 12(4). p. 388-422.
- J13. Tovi Grossman, Ravin Balakrishnan. (2005). A probabilistic approach to modeling two-dimensional pointing. *ACM Transactions on Computer-Human Interaction*, 12(3). p. 435-459.
- J14. Anastasia Bezerianos, Ravin Balakrishnan. (2005). Canvas portals: view and space management on large displays. *IEEE Computer Graphics and Applications* 25(4). p. 34-43.
- J15. Ravin Balakrishnan. (2004). "Beating" Fitts' law: Virtual enhancements for pointing facilitation. *International Journal of Human-Computer Studies*, 61(6). p. 857-874.
- J16. George W. Fitzmaurice, Azam Khan, William Buxton, Gordon Kurtenbach, Ravin Balakrishnan. (2003). Sentient data access via a diverse society of devices. *ACM Queue*, November 2003. p. 53-62.
- J17. Ravin Balakrishnan, George W. Fitzmaurice, Gordon Kurtenbach. (2001). User interfaces for volumetric displays. *IEEE Computer*, March 2001. p. 37-45.
- J18. William Buxton, George W. Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach. (2000). Large displays in automotive design. *IEEE Computer Graphics and Applications*, July/August 2000. p. 68-75.
- J19. George W. Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach. (1999). Sampling, synthesis, and input devices. *Communications of the ACM*, August 1999, 42(8). p. 54-63.
- J20. Colin Ware, Ravin Balakrishnan. (1994). Reaching for objects in VR displays: Lag and frame rate. *ACM Transactions on Computer-Human Interaction*, 1(4). p. 331-356.

Refereed conference full-length papers

- C1. Jiannan Li, Maurício Sousa, Chu Li, Jessie Liu, Yan Chen, Ravin Balakrishnan, Tovi Grossman. (2022). Asteroids: Exploring Swarms of Mini-Telepresence Robots for Physical Skill Demonstration. *Proceedings of CHI 2022 – the ACM Conference on Human Factors in Computing Systems*.
- C2. Jiannan Li, Jiahe Lyu, Maurício Sousa, Ravin Balakrishnan, Anthony Tang, Tovi Grossman. (2021). Route Tapestries: Navigating 360° Virtual Tour Videos Using Slit-Scan Visualizations. *Proceedings of UIST 2021 – the ACM Symposium on User Interface Software and Technology*.

- C3. Jiannan Li, Ravin Balakrishnan, Tovi Grossman. (2020). StarHopper: A Touch Interface for Remote Object-Centric Drone Navigation *Proceedings of Graphics Interface 2020*.
- C4. Laura Di Chen, Ravin Balakrishnan, Tovi Grossman (2020). Disambiguation Techniques for Freehand Object Manipulations in Virtual Reality. *Proceedings of IEEE Conference on Virtual Reality and 3D User Interfaces*.
- C5. Di Chen, Dustin Freeman, Ravin Balakrishnan. (2019). Integrating Multimedia Tools to Enrich Interactions in Live Streaming for Language Learning. *Proceedings of CHI 2019 – the ACM Conference on Human Factors in Computing Systems*.
- C6. Aakar Gupta, Jiushan Yang, Ravin Balakrishnan. (2018). Asterisk and Obelisk: Motion Codes for Passive Tagging. *Proceedings of UIST 2018 – the ACM Symposium on User Interface Software and Technology*.
- C7. Teng Han, Jiannan Li, Khalad Hasan, Keisuke Nakamura, Randy Gomez, Ravin Balakrishnan, Pourang Irani. (2018). PageFlip: Leveraging page-flipping gestures for efficient command and value selection on smartwatches. *Proceedings of CHI 2018 – the ACM Conference on Human Factors in Computing Systems*.
- C8. Aakar Gupta, Antony Irudayaraj, Ravin Balakrishnan. (2017). HapticClench: Investigating squeeze sensations using memory alloys. *Proceedings of UIST 2017 – the ACM Symposium on User Interface Software and Technology*.
- C9. Aakar Gupta, Thomas Pietrzak, Cleon Yau, Nicolas Roussel, Ravin Balakrishnan. (2017). Summon and Select: Rapid interaction with interface controls in mid-air. *Proceedings of ISS 2017 – the ACM Conference on Interactive Surfaces and Spaces*.
- C10. Rorik Henrikson, Bruno De Araujo, Fanny Chevalier, Karan Singh, Ravin Balakrishnan. (2016). Multi-device storyboards for cinematic narratives in VR. *Proceedings of UIST 2016 – the ACM Symposium on User Interface Software and Technology*.
- C11. Aakar Gupta, Antony Irudayaraj, Vimal Chandran, Goutham Palaniappan, Khai Truong, Ravin Balakrishnan. (2016). Haptic learning of freehand semaphoric gesture shortcuts. *Proceedings of UIST 2016 – the ACM Symposium on User Interface Software and Technology*.
- C12. Aakar Gupta, Muhammed Anwar, Ravin Balakrishnan. (2016). Porous interfaces for small screen multitasking using finger identification. *Proceedings of UIST 2016 – the ACM Symposium on User Interface Software and Technology*.
- C13. Dustin Freeman, Ravin Balakrishnan. (2016). Improv Remix: Mixed-reality video manipulation using whole-body interaction to extend improvised theatre. *Proceedings of DIS 2016 – The ACM Conference on Designing Interactive Systems*.
- C14. Rorik Henrikson, Bruno De Araujo, Fanny Chevalier, Karan Singh, Ravin Balakrishnan. (2016) Storeboard: Sketching stereoscopic storyboards. *Proceedings of CHI 2016 – the ACM Conference on Human Factors in Computing Systems*. p. 4587–4598
- C15. Aakar Gupta, Ravin Balakrishnan. (2016). DualKey: Miniature screen text entry via finger manipulation. *Proceedings of CHI 2016 – the ACM Conference on Human Factors in Computing Systems*. p. 59-70 [**ACM CHI 2016 Best Paper Honorable Mention Award**]
- C16. Aakar Gupta, Thomas Pietrzak, Nicolas Roussel, Ravin Balakrishnan. (2016). Direct manipulation in tactile displays. *Proceedings of CHI 2016 – the ACM Conference on Human Factors in Computing Systems*. p. 3683–3693. [**ACM CHI 2016 Best Paper Honorable Mention Award**]
- C17. Dustin Freeman, Stephanie Santosa, Fanny Chevalier, Ravin Balakrishnan, Karan Singh. (2013). LACES: Live authoring through compositing and editing of streaming video. *Proceedings of CHI 2014 – the ACM Conference on Human Factors in Computing Systems*. p. 1207-1216.

- C18. Jian Zhao, Daniel Wigdor, Ravin Balakrishnan. (2013). TrailMap: Facilitating information seeking in a multi-scale digital map via implicit bookmarking. *Proceedings of CHI 2013 – the ACM Conference on Human Factors in Computing Systems*. p. 3009-3018.
- C19. Sajad Shirali-Shahreza, Gerald Penn, Ravin Balakrishnan, Yashar Ganjali. (2013). SeeSay and HearSay CAPTCHAs for mobile interaction. *Proceedings of CHI 2013 – the ACM Conference on Human Factors in Computing Systems*. p. 2147-2156.
- C20. Stephanie Santosa, Fanny Chevalier, Ravin Balakrishnan, Karan Singh. (2013). Direct space-time trajectory control for visual media editing. *Proceedings of CHI 2013 – the ACM Conference on Human Factors in Computing Systems*. p. 1149-1158. **[ACM CHI 2013 Best Paper Honorable Mention Award]**
- C21. Mohit Jain, Ravin Balakrishnan. (2012). User learning and performance with bezel menus. *Proceedings of CHI 2012 – the ACM Conference on Human Factors in Computing Systems*. p. 2221-2230.
- C22. Aakar Gupta, William Thies, Edward Cutrell, Ravin Balakrishnan. (2012). mClerk: Enabling mobile crowdsourcing in developing regions. *Proceedings of CHI 2012 – the ACM Conference on Human Factors in Computing Systems*. p. 1843-1852.
- C23. Navkar Samdaria, Akhil Mathur, Ravin Balakrishnan. (2012). Paying in kind for crowdsourced work in developing regions. *Proceedings of the Pervasive 2012 Conference*. p. 343-360.
- C24. Dustin Freeman, Ravin Balakrishnan. (2011). Tangible actions. *Proceedings of ITS 2011 – the ACM International Conference on Interactive Tabletops and Surfaces*. p. 87-96.
- C25. Yannick Thiel, Karan Singh, Ravin Balakrishnan. (2011). Elasticurves: Exploiting stroke dynamics and inertia for the real-time neatening of sketched 2D curves. *Proceedings of UIST 2011 – the ACM Symposium on User Interface Software and Technology*. p. 383-392.
- C26. Sajad Shirali-Shahreza, Yashar Ganjali, Ravin Balakrishnan. (2011). Verifying human users in speech-based interactions. *Proceedings of the Annual Conference of the International Speech Communication Association (INTERSPEECH)*. p. 1585-1588.
- C27. Akhil Mathur, Divya Ramachandran, Ed Cutrell, Ravin Balakrishnan. (2011). An exploratory study on the use of camera phones and pico projectors in rural India. *Proceedings of MobileHCI 2011*. p. 347-356.
- C28. Mohit Jain, Jeremy Birnholtz, Ed Cutrell, Ravin Balakrishnan. (2011). Exploring display techniques for mobile collaborative learning in developing regions. *Proceedings of MobileHCI 2011*. p. 81-90.
- C29. Jian Zhao, Fanny Chevalier, Ravin Balakrishnan. (2011). KronoMiner: Using multi-foci navigation for the visual exploration of time-series data. *Proceedings of CHI 2011 – the ACM Conference on Human Factors in Computing Systems*. p. 1737-1746.
- C30. Chun Yu, Yuanchun Shi, Ravin Balakrishnan, Xiangliang Meng, Yue Suo, Mingming Fan, Yongqiang Qin. (2010). The satellite cursor: achieving MAGIC pointing without gaze tracking using multiple cursors. *Proceedings of UIST 2010 – the ACM Symposium on User Interface Software and Technology*. p. 163-172.
- C31. James Scott, Shahram Izadi, Leila Rezai, Dominika Ruskowski, Xiaojun Bi, Ravin Balakrishnan. (2010). RearType: Text entry using keys on the back of a device. *Proceedings of MobileHCI 2010*. p. 171-179.
- C32. Abhishek Ranjan, Jeremy Birnholtz, Rorik Henrikson, Ravin Balakrishnan, Dana Lee. (2010). Automatic camera control using unobtrusive vision and audio tracking. *Proceedings of GI 2010 – the Graphics Interface Conference*. p. 47-54.
- C33. Daniel Vogel, Ravin Balakrishnan. (2010). Occlusion-aware interfaces. *Proceedings of CHI 2010 – the ACM Conference on Human Factors in Computing Systems*. p. 263-272. **[ACM CHI 2010 Best Paper Award]**

- C34. Patrick Dubroy, Ravin Balakrishnan. (2010). A study of tabbed browsing among Mozilla Firefox users . *Proceedings of CHI 2010 – the ACM Conference on Human Factors in Computing Systems*. p. 673-682.
- C35. Xiaojun Bi, Seok-Hyung Bae, Ravin Balakrishnan. (2010). Effects of interior bezels of tiled-monitor large displays on visual search, tunnel steering, and target selection . *Proceedings of CHI 2010 – the ACM Conference on Human Factors in Computing Systems*. p. 65-74.
- C36. Saleema Amershi, Meredith Ringel Morris, Neema Moraveji, Ravin Balakrishnan, Kentaro Toyama. (2010). Multiple mouse text entry for single-display groupware. *Proceedings of CSCW 2010 – the ACM Conference on Computer Supported Cooperative Work*. p. 169-178. [**ACM CSCW 2010 Best Paper Honorable Mention Award**]
- C37. Seok-Hyung Bae, Ravin Balakrishnan, Karan Singh. (2009). EverybodyLovesSketch: 3D sketching for a broader audience. *Proceedings of UIST 2009 – the ACM Symposium on User Interface Software and Technology*. p. 59-68.
- C38. Scott Saponas, Desney Tan, Dan Morris, Ravin Balakrishnan, Jim Turner, James Landay. (2009). Enabling always-available input with muscle-computer interfaces. *Proceedings of UIST 2009 – the ACM Symposium on User Interface Software and Technology*. p. 167-176.
- C39. Gursharan Singh, Leah Findlater, Kentaro Toyama, Scott Helmer, Rikin Gandhi, Ravin Balakrishnan. Numeric paper forms for NGOs. (2009). *Proceedings of ICTD 2009 – the IEEE/ACM International Conference on Information and Communication Technologies and Development*. p. 406-416.
- C40. Gerry Chu, Sambit Satpathy, Kentaro Toyama, Rikin Gandhi, Ravin Balakrishnan. (2009). Featherweight multimedia for information dissemination. *Proceedings of ICTD 2009 – the IEEE/ACM International Conference on Information and Communication Technologies and Development*. p. 337-347.
- C41. Daniel Vogel, Matthew Cudmore, Gery Casiez, Ravin Balakrishnan, Liam Keliher. (2009). Hand occlusion with tablet-sized direct pen input. *Proceedings of CHI 2009 – the ACM Conference on Human Factors in Computing Systems*. p. 557-566.
- C42. Xiaojun Bi, Ravin Balakrishnan. (2009). Comparing usage of a large high-resolution display to single or dual desktop displays for daily work. *Proceedings of CHI 2009 – the ACM Conference on Human Factors in Computing Systems*. p. 1005-1014.
- C43. Clifton Forlines, Ravin Balakrishnan. (2009). Improving visual search with image segmentation. *Proceedings of CHI 2009 – the ACM Conference on Human Factors in Computing Systems*. p. 1093-1102. [**ACM CHI 2009 Best Paper Nominee**]
- C44. Leah Findlater, Ravin Balakrishnan, Kentaro Toyama. (2009). Comparing semiliterate and illiterate users' ability to transition from audio+text to text-only interaction. *Proceedings of CHI 2009 – the ACM Conference on Human Factors in Computing Systems*. p. 1751-1760.
- C45. Ken Hinckley, Morgan Dixon, Raman Sarin, Francois Guimbretiere, Ravin Balakrishnan. (2009). Codex: A dual screen computer. *Proceedings of CHI 2009 – the ACM Conference on Human Factors in Computing Systems*. p. 1933-1942.
- C46. Neema Moraveji, Kori Inkpen, Ed Cutrell, Ravin Balakrishnan. (2009). A mischief of mice: Examining children's performance in single display groupware systems with 1 to 32 mice. *Proceedings of CHI 2009 – the ACM Conference on Human Factors in Computing Systems*. p.2157-2166.
- C47. Xiang Cao, Michael Massimi, Ravin Balakrishnan. (2008). Flashlight Jigsaw: An exploratory study of an ad-hoc multi-player game on public displays. *Proceedings of CSCW 2008 – the ACM Conference on Computer Supported Cooperative Work*. p. 77-86.

- C48. Hrvoje Benko, Andrew Wilson, Ravin Balakrishnan. (2008). Sphere: Multi-touch interactions on a spherical display. *Proceedings of UIST 2008 – the ACM Symposium on User Interface Software and Technology*. p. 77-86.
- C49. Xiaojun Bi, Tomer Moscovich, Gonzalo Ramos, Ravin Balakrishnan, Ken Hinckley. (2008). An exploration of pen rolling for pen-based interaction. *Proceedings of UIST 2008 – the ACM Symposium on User Interface Software and Technology*. p. 191-200.
- C50. Seok-Hyung Bae, Ravin Balakrishnan, Karan Singh. (2008). ILoveSketch: As-natural-as-possible sketching system for creating 3D curve models. *Proceedings of UIST 2008 – the ACM Symposium on User Interface Software and Technology*. p. 151-160.
- C51. Xiang Cao, Andrew Wilson, Ravin Balakrishnan, Ken Hinckley, Scott E. Hudson. (2008). ShapeTouch: Leveraging contact shape on interactive surfaces. *Proceedings of TABLETOP 2008 – the IEEE International Workshop on Horizontal Interactive Human Computer Systems*. p. 139-146.
- C52. Archana Prasad, Indrani Medhi, Kentaro Toyama, Ravin Balakrishnan. (2008). Exploring the feasibility of video mail for illiterate users. *Proceedings of AVI 2008 - the Conference on Advanced Visual Interfaces*. p. 103-110.
- C53. Ryan Schmidt, Karan Singh, Ravin Balakrishnan. (2008). Sketching and composing widgets for 3D manipulation. *Computer Graphics Forum*, 27(2), *Proceedings of Eurographics Conference*. p. 301-310.
- C54. Scott Saponas, Desney Tan, Dan Morris, Ravin Balakrishnan. (2008). Demonstrating the feasibility of using forearm electromyography for muscle-computer interfaces. *Proceedings of CHI 2008 – the ACM Conference on Human Factors in Computing Systems*. p. 515-524.
- C55. Xiang Cao, Jacky Jie Li, Ravin Balakrishnan. (2008). Peephole pointing: Modeling acquisition of dynamically revealed targets. *Proceedings of CHI 2008 – the ACM Conference on Human Factors in Computing Systems*. p. 1699-1708. [**ACM CHI 2008 Best Paper Nominee**]
- C56. Tovi Grossman, Ravin Balakrishnan. (2008). Collaborative interaction with volumetric displays. *Proceedings of CHI 2008 – the ACM Conference on Human Factors in Computing Systems*. p. 383-392.
- C57. Clifton Forlines, Ravin Balakrishnan. (2008). Evaluating tactile feedback and direct vs. indirect stylus input in pointing and crossing selection tasks. *Proceedings of CHI 2008 – the ACM Conference on Human Factors in Computing Systems*. p. 1563-1572.
- C58. Abhishek Ranjan, Jeremy Birnholtz, Ravin Balakrishnan. (2008). Improving meeting capture by applying television production principles with audio and motion detection. *Proceedings of CHI 2008 – the ACM Conference on Human Factors in Computing Systems*. p. 227-238.
- C59. Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowicz, Derek Nowrouzezahrai, Ravin Balakrishnan, Karan Singh. (2008). Video browsing by direct manipulation. *Proceedings of CHI 2008 – the ACM Conference on Human Factors in Computing Systems*. p. 237-246.
- C60. Xiang Cao, Clifton Forlines, Ravin Balakrishnan. (2007). Multi-user interaction using handheld projectors. *Proceedings of UIST 2007 – the ACM Symposium on User Interface Software and Technology*. p. 43-52.
- C61. Shengdong Zhao, Pierre Dragicevic, Mark Chignell, Ravin Balakrishnan, Patrick Baudisch. (2007). earPod: Eyes-free menu selection using touch input and reactive audio feedback. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 1395-1404.
- C62. Clifton Forlines, Chia Shen, Daniel Wigdor, Ravin Balakrishnan. (2007). Direct-touch vs. mouse input for tabletop displays. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 647-656.

- C63. Gonzalo Ramos, Ravin Balakrishnan. (2007). Pressure marks. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 1375-1384.
- C64. Gonzalo Ramos, Andy Cockburn, Ravin Balakrishnan, Michel Beaudouin-Lafon. (2007). Pointing lenses. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 757-766.
- C65. Abhishek Ranjan, Jeremy Birnholtz, Ravin Balakrishnan. (2007). Dynamic shared visual spaces: Experimenting with automatic camera control in a remote repair task. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 1177-1186.
- C66. Daniel Wigdor, Chia Shen, Clifton Forlines, Ravin Balakrishnan. (2007). Perception of elementary graphical elements in tabletop and multi-surface environments. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 473-482.
- C67. Jeremy Birnholtz, Tovi Grossman, Clarissa Mak, Ravin Balakrishnan. (2007). An exploratory study of input configuration and group process in a negotiation task using a large display. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 91-100.
- C68. Tovi Grossman, Nicholas Kong, Ravin Balakrishnan. (2007). Modeling pointing at targets of arbitrary shapes. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 463-472.
- C69. Tovi Grossman, Daniel Wigdor, Ravin Balakrishnan. (2007). Exploring and reducing the effects of orientation on text readability in volumetric displays. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 483-492.
- C70. Tovi Grossman, Pierre Dragicevic, Ravin Balakrishnan. (2007). Strategies for accelerating on-line learning of hotkeys. *Proceedings of CHI 2007 – the ACM Conference on Human Factors in Computing Systems*. p. 1591-1600.
- C71. Jia Sheng, Ravin Balakrishnan, Karan Singh. (2006). An interface for 3D sculpting via physical proxy. *Proceedings of GRAPHITE 2006 – International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*. p. 213-220.
- C72. Abhishek Ranjan, Jeremy Birnholtz, Ravin Balakrishnan. (2006). An exploratory analysis of partner action and camera control in a video-mediated collaborative task. *Proceedings of CSCW 2006 – the ACM Conference on Computer Supported Cooperative Work*. p. 403-412. **[ACM CSCW 2006 Best Paper Honorable Mention Award]**
- C73. Clifton Forlines, Chia Shen, Daniel Wigdor, Ravin Balakrishnan. (2006). Exploring the effects of group size and display configuration on visual search. *Proceedings of CSCW 2006 – the ACM Conference on Computer Supported Cooperative Work*. p. 11-20.
- C74. Xiang Cao, Ravin Balakrishnan. (2006). Interacting with dynamically defined information spaces using a handheld projector and a pen. *Proceedings of UIST 2006 – the ACM Symposium on User Interface Software and Technology*. p. 225-234.
- C75. Daniel Wigdor, Darren Leigh, Clifton Forlines, Samuel Shipman, John Barnwell, Ravin Balakrishnan, Chia Shen. (2006). Under the table interaction. *Proceedings of UIST 2006 – the ACM Symposium on User Interface Software and Technology*. p. 259-268.
- C76. Anastasia Bezerianos, Pierre Dragicevic, Ravin Balakrishnan. (2006). Mnemonic rendering: An image-based approach for exposing hidden changes in dynamic displays. *Proceedings of UIST 2006 – the ACM Symposium on User Interface Software and Technology*. p. 159-168. **[ACM UIST 2006 Best Student Paper Award, and one of 3 papers invited to a UIST reprise session at ACM SIGGRAPH 2007]**

- C77. Tovi Grossman, Ravin Balakrishnan. (2006). The design and evaluation of selection techniques for 3d volumetric displays. *Proceedings of UIST 2006 – the ACM Symposium on User Interface Software and Technology*. p. 3-12.
- C78. Clifton Forlines, Daniel Vogel, Ravin Balakrishnan. (2006). HybridPointing: Fluid switching between absolute and relative pointing with a direct input device. *Proceedings of UIST 2006 – the ACM Symposium on User Interface Software and Technology*. p. 211-220.
- C79. Tovi Grossman, Ravin Balakrishnan. (2006). An evaluation of depth perception on volumetric displays. *Proceedings of AVI 2006 – the Conference on Advanced Visual Interfaces*. p. 193-200.
- C80. Anand Agarawala, Ravin Balakrishnan. (2006). Keepin' it real: Pushing the desktop metaphor with physics, piles and the pen. *Proceedings of CHI 2006 – the ACM Conference on Human Factors in Computing Systems*. p. 1283-1292.
- C81. Daniel Wigdor, Chia Shen, Clifton Forlines, Ravin Balakrishnan. (2006). Effects of display position and control space orientation on user preference and performance. *Proceedings of CHI 2006 – the ACM Conference on Human Factors in Computing Systems*. p. 309-318.
- C82. Tovi Grossman, Ken Hinckley, Patrick Baudisch, Maneesh Agrawala, Ravin Balakrishnan. (2006). Hover widgets: Using the tracking state to extend the capabilities of pen-operated devices. *Proceedings of CHI 2006 – the ACM Conference on Human Factors in Computing Systems*. p. 861-870.
- C83. Abhishek Ranjan, Ravin Balakrishnan, Mark Chignell. (2006). Searching in audio: The utility of transcripts, dichotic presentation, and time-compression. *Proceedings of CHI 2006 – the ACM Conference on Human Factors in Computing Systems*. p. 721-730.
- C84. Michael Wu, Chia Shen, Kathy Ryall, Clifton Forlines, Ravin Balakrishnan. (2006). Gesture registration, relaxation, and reuse for multi-point direct-touch surfaces. *Proceedings of TableTop 2006 – The IEEE International Workshop on Horizontal Interactive Human Computer Systems*. p. 183-190.
- C85. Daniel Vogel, Ravin Balakrishnan. (2005). Distant freehand pointing and clicking on very large high resolution displays. *Proceedings of UIST 2005 – the ACM Symposium on User Interface Software and Technology*. p. 33-42.
- C86. Shahzad Malik, Abhishek Ranjan, Ravin Balakrishnan. (2005). Interacting with large displays from a distance with vision-tracked multi-finger gestural input. *Proceedings of UIST 2005 – the ACM Symposium on User Interface Software and Technology*. p. 43-52. **[One of 3 papers invited to a UIST reprise session at ACM SIGGRAPH 2006]**
- C87. Gonzalo Ramos, Ravin Balakrishnan. (2005). Zliding: Fluid zooming and sliding for high precision parameter manipulation. *Proceedings of UIST 2005 – the ACM Symposium on User Interface Software and Technology*. p. 143-152.
- C88. Clifton Forlines, Ravin Balakrishnan, Paul Beardsley, Jeroen van Baar, Ramesh Raskar. (2005). Zoom-and-Pick: Facilitating visual zooming and precision pointing with interactive handheld projectors. *Proceedings of UIST 2005 – the ACM Symposium on User Interface Software and Technology*. p. 73-82.
- C89. Michael McGuffin, Ravin Balakrishnan. (2005). Interactive visualization of genealogical graphs. *Proceedings of InfoVis 2005 – the IEEE Symposium on Information Visualization*. p. 17-24.
- C90. Theophanis Tsandilas, Ravin Balakrishnan. (2005). An evaluation of techniques for reducing spatial interference in single display groupware. *Proceedings of ECSCW 2005 – the European Conference on Computer Supported Cooperative Work*. p. 221-240.
- C91. Daniel Wigdor, Ravin Balakrishnan. (2005). Empirical investigation into the effect of orientation on text readability in tabletop displays. *Proceedings of ECSCW 2005 – the European Conference on Computer Supported Cooperative Work*. p. 205-224.

- C92. Xiang Cao, Ravin Balakrishnan. (2005). Evaluation of an online adaptive gesture interface with command prediction. *Proceedings of GI 2005 – the Graphics Interface Conference*. p. 187-194. **[GI 2005 Best Student Paper Award]**
- C93. Tovi Grossman, Ravin Balakrishnan. (2005). The Bubble Cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. *Proceedings of CHI 2005 – the ACM Conference on Human Factors in Computing Systems*. p. 281-290. **[ACM CHI 2005 Best Paper Award]**
- C94. Anastasia Bezerianos, Ravin Balakrishnan. (2005). The Vacuum: Facilitating the manipulation of distant objects. *Proceedings of CHI 2005 – the ACM Conference on Human Factors in Computing Systems*. p. 361-370.
- C95. Daniel Vogel, Ravin Balakrishnan. (2004). Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users. *Proceedings of UIST 2004 – the ACM Symposium on User Interface Software and Technology*. p. 137-146.
- C96. Shengdong Zhao, Ravin Balakrishnan. (2004). Simple vs. compound mark hierarchical marking menus. *Proceedings of UIST 2004 – the ACM Symposium on User Interface Software and Technology*. p. 33-42.
- C97. Tovi Grossman, Daniel Wigdor, Ravin Balakrishnan. (2004). Multi-finger gestural interaction with 3D volumetric displays. *Proceedings of UIST 2004 – the ACM Symposium on User Interface Software and Technology*. p. 61-70. **[ACM UIST 2004 Best Paper Award, and one of 3 papers invited to a UIST reprise session at ACM SIGGRAPH 2005]**.
- C98. Karan Singh, Ravin Balakrishnan. (2004). Visualizing 3D scenes using non-linear projections and data mining of previous camera movements. *Proceedings of Afrigraph 2004 – the ACM African Graphics Conference*. p. 41-48.
- C99. Michael McGuffin, Gord Davison, Ravin Balakrishnan. (2004). Expand-ahead: A space-filling strategy for browsing trees affording faster drill-down. *Proceedings of InfoVis 2004 – the IEEE Symposium on Information Visualization*. p. 119-126.
- C100. Wael Aboelsaadat, Ravin Balakrishnan. (2004). An empirical comparison of transparency on one and two layer displays. *Proceedings of HCI 2004 – the British HCI Conference*. p. 53-67.
- C101. Steven Tsang, Ravin Balakrishnan, Karan Singh, Abhishek Ranjan. (2004). A suggestive interface for image guided 3D sketching. *Proceedings of CHI 2004 – the ACM Conference on Human Factors in Computing Systems*. p. 591-598.
- C102. Gonzalo Ramos, Matthew Boulos, Ravin Balakrishnan. (2004). Pressure widgets. *Proceedings of CHI 2004 – the ACM Conference on Human Factors in Computing Systems*. p. 487-494.
- C103. Tovi Grossman, Ravin Balakrishnan. (2004). Pointing at trivariate targets in 3D environments. *Proceedings of CHI 2004 – the ACM Conference on Human Factors in Computing Systems*. p. 447-454.
- C104. Daniel Wigdor, Ravin Balakrishnan. (2004). A comparison of consecutive and concurrent input text entry techniques for mobile phones. *Proceedings of CHI 2004 – the ACM Conference on Human Factors in Computing Systems*. p. 81-88.
- C105. Michael Wu, Ravin Balakrishnan. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *Proceedings of UIST 2003 – the ACM Symposium on User Interface Software and Technology*. p. 193-202.
- C106. Daniel Wigdor, Ravin Balakrishnan. (2003). TiltText: Using tilt for text input to mobile phones. *Proceedings of UIST 2003 – the ACM Symposium on User Interface Software and Technology*. p. 81-90.
- C107. Gonzalo Ramos, Ravin Balakrishnan. (2003). Fluid interaction techniques for the control and annotation of digital video. *Proceedings of UIST 2003 – the ACM Symposium on User Interface Software and Technology*. p. 105-114.
- C108. Xiang Cao, Ravin Balakrishnan. (2003). VisionWand: Interaction techniques for large displays using a passive wand tracked in 3D. *Proceedings of UIST 2003 – the ACM Symposium on User Interface Software and Technology*. p. 173-182. **[One of 3 papers invited to a “Best of UIST” reprise session at ACM SIGGRAPH 2004]**

- C109. Michael McGuffin, Liviu Tancau, Ravin Balakrishnan. (2003). Using deformations for browsing volumetric data. *Proceedings of Vis 2003 – the IEEE Visualization Conference*. p. 401-408.
- C110. Tovi Grossman, Ravin Balakrishnan, Karan Singh. (2003). An interface for creating and manipulating curves using a high degree-of-freedom input device. *Proceedings of CHI 2003 – the ACM Conference on Human Factors in Computing Systems*. p. 185-192.
- C111. Nicholas Burtnyk, Azam Khan, George W. Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach. (2002). StyleCam: Interactive stylized 3D navigation using integrated spatial & temporal controls. *Proceedings of UIST 2002 – the ACM Symposium on User Interface Software & Technology*. p. 101-110.
- C112. Michael McGuffin, Ravin Balakrishnan. (2002). Acquisition of expanding targets. *Proceedings of CHI 2002 – the ACM Conference on Human Factors in Computing Systems*. p. 57-64.
- C113. Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George W. Fitzmaurice, Azam Khan, William Buxton. (2002). Creating principal 3D curves with digital tape drawing. *Proceedings of CHI 2002 – the ACM Conference on Human Factors in Computing Systems*. p. 121-128.
- C114. Tovi Grossman, Ravin Balakrishnan, Gordon Kurtenbach, George W. Fitzmaurice, Azam Khan, William Buxton. (2001). Interaction techniques for 3D modeling on large displays. *Proceedings of I3D 2001 – the ACM Symposium on Interactive 3D Graphics*. p. 17-23.
- C115. Ravin Balakrishnan, Ken Hinckley. (2000). Symmetric bimanual interaction. *Proceedings of CHI 2000 – the ACM Conference on Human Factors in Computing Systems*. p. 33-40.
- C116. Ravin Balakrishnan, Ken Hinckley. (1999). The role of kinesthetic reference frames in two-handed input performance. *Proceedings of UIST 1999 – the ACM Symposium on User Interface Software & Technology*. p. 171-178.
- C117. Ravin Balakrishnan, George W. Fitzmaurice, Gordon Kurtenbach, William Buxton. (1999). Digital tape drawing. *Proceedings of UIST 1999 – the ACM Symposium on User Interface Software & Technology*. p. 161-169.
- C118. Ravin Balakrishnan, Gordon Kurtenbach. (1999). Exploring bimanual camera control and object manipulation in 3D graphics interfaces. *Proceedings of CHI 1999 – the ACM Conference on Human Factors in Computing Systems*. p. 53-63.
- C119. George W. Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach, William Buxton. (1999). An exploration into supporting artwork orientation in the user interface. *Proceedings of CHI 1999 – the ACM Conference on Human Factors in Computing Systems*. p. 167-174.
- C120. Ravin Balakrishnan, George W. Fitzmaurice, Gordon Kurtenbach, Karan Singh. (1999). Exploring interactive curve and surface manipulation using a bend and twist sensitive input strip. *Proceedings of I3D 1999 – the ACM Symposium on Interactive 3D Graphics*. p. 111-118.
- C121. Ravin Balakrishnan, Pranay Patel. (1998). The PadMouse: Facilitating selection and spatial positioning for the non-dominant hand. *Proceedings of CHI 1998 – the ACM Conference on Human Factors in Computing Systems*. p. 9-16.
- C122. Ravin Balakrishnan, Thomas Baudel, Gordon Kurtenbach, George W. Fitzmaurice. (1997). The Rockin'Mouse: Integral 3D manipulation on a plane. *Proceedings of CHI 1997 – the ACM Conference on Human Factors in Computing Systems*. p. 311-318.
- C123. Ravin Balakrishnan, I. Scott MacKenzie. (1997). Performance differences in the fingers, wrist, and forearm in computer input control. *Proceedings of CHI 1997 – the ACM Conference on Human Factors in Computing Systems*. p. 303-310.
- C124. Colin Ware, Ravin Balakrishnan. (1994). Target acquisition in fishtank VR: The effects of lag and frame rate. *Proceedings of the Graphics Interface Conference*. p. 1-7.

Refereed conference notes and short papers

- N1. Gerry Chu, Tomer Moscovich, Ravin Balakrishnan. (2009). Haptic conviction widgets. *Proceedings of GI 2009 – the Graphics Interface Conference*. p. 207-210.
- N2. Daniel Wigdor, Chia Shen, Clifton Forlines, Ravin Balakrishnan. (2006). Table-centric interactive spaces for real-time collaboration. *Proceedings of AVI 2006 – the Conference on Advanced Visual Interfaces*. p. 103-107.
- N3. Michael Tsang, Nigel Morris, Ravin Balakrishnan. (2004). Temporal thumbnails: rapid visualization of time-based viewing data. *Proceedings of AVI 2004 – the Conference on Advanced Visual Interfaces*. p. 175-178.

Conference abstracts, posters and demos

1. Jeremy Birnholtz, Abhishek Ranjan, Ravin Balakrishnan. (2008). Error and coupling: Extending common ground to improve the provision of visual information for collaborative tasks. *Conference of the International Communication Association, Montreal, Canada, May 22-26, 2008*.
2. Jeremy Birnholtz, Abhishek Ranjan, Ravin Balakrishnan. (2007). Using motion tracking data to augment video recordings in experimental social science research. *Proceedings of the Third International Conference on e-Social Science*.
3. Igor Mordatch, Patrick Coleman, Karan Singh, Ravin Balakrishnan. (2007). Interface techniques for 3D control of spatial keyframing. *Poster in ACM SIGGRAPH Conference*.
4. Daniel Wigdor, Chia Shen, Clifton Forlines, Ravin Balakrishnan. (2006). Table-centric interactive spaces for real-time collaboration: a video demonstration. *Video Proceedings of CSCW 2006 – the ACM Conference on Computer Supported Cooperative Work*.
5. Daniel Wigdor, Chia Shen, Clifton Forlines, Ravin Balakrishnan. (2006). Table-centric interactive spaces for real-time collaboration: Solutions, evaluation, and application scenarios. *Proceedings of CollabTech 2006*.
6. Ravin Balakrishnan, Colin Ware, Tim Smith. (1994). Virtual handtool with force-feedback. *Companion Proceedings of CHI 1994 – the ACM Conference on Human Factors in Computing Systems*. p. 83-84.

Technical Reports

1. Gonzalo Ramos, Ravin Balakrishnan. (2004). Visual features and interference in pressure widgets. Technical Report, DGP Lab, University of Toronto, DGP-TR-2004-003.
2. Anastasia Bezerianos, Ravin Balakrishnan. (2004). Interaction and visualization techniques for very large scale high resolution displays. Technical Report, DGP Lab, University of Toronto, DGP-TR-2004-002.

Theses

1. Ravin Balakrishnan. (2001). Issues in bimanual interaction for computer graphics. Ph.D. Thesis, Department of Computer Science, University of Toronto. February 2001.
2. Ravin Balakrishnan. (1997). The evolution and evaluation of a 3D input device. M.Sc. Thesis, Department of Computer Science, University of Toronto. January 1997.

Patents

1. Hrvoje Benko, Andrew Wilson, Billy Chen, Ravin Balakrishnan, Patrick Baudisch. (2015). Touch interaction with a curved display. *United States Patent #. 9,218,116*
2. Desney Tan, Daniel Morris, Scott Saponas, Ravin Balakrishnan. (2014). Recognizing finger gestures from forearm EMG signals. *United States Patent #. 8,892,479*
3. Desney Tan, Daniel Morris, Scott Saponas, Ravin Balakrishnan. (2013). Recognizing gestures from forearm EMG signals. *United States Patent #. 8,447,704*

4. Anand Agarawala, Ravin Balakrishnan. (2013). System for organizing and visualizing display objects. *United States Patent # 8,402,382*
5. Steve Tsang, Karan Singh, Abhishek Ranjan, Ravin Balakrishnan. (2012). Method, system and computer program for using a suggestive modeling interface. *United States Patent # 8,300,062*
6. Kentaro Toyama, Gerald Chu, Ravin Balakrishnan. (2012). Talking paper authoring tools. *United States Patent # 8,201,074*.
7. Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice. (2010). System for physical rotation of volumetric display enclosures to facilitate viewing. *United States Patent # 7724251*.
8. Gordon Kurtenbach, George Fitzmaurice, Ravin Balakrishnan. (2010). Volume management system for volumetric displays. *United States Patent # 7839400, 7,986,318*
9. Nicolas Burtnyk, Azam Khan, George Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach. (2009). System and method of integrated spatial and temporal navigation. *United States Patent # 7589732*.
10. George Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach. (2009). Widgets displayed and operable on a surface of a volumetric display enclosure. *United States Patent # 7554541*.
11. Clifton Forlines, Ravin Balakrishnan. (2009). Method and system for switching between absolute and relative pointing with direct input devices. *United States Patent # 7640518*.
12. Clifton Forlines, Ravin Balakrishnan. (2009). Method for stabilizing and precisely locating pointers generated by handheld direct pointing devices. *United States Patent #7486274*.
13. Ravin Balakrishnan, George Fitzmaurice, Gordon Kurtenbach. (2008). Techniques for pointing to locations within a volumetric display. *United States Patent # 7324085, 7,528,823*
14. Ravin Balakrishnan, George Fitzmaurice, Gordon Kurtenbach, Karan Singh. (2008). System for creating and modifying curves and surfaces. *United States Patent #7289121*
15. George Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach. (2007). Graphical user interface widgets viewable and readable from multiple viewpoints in a volumetric display. *United States Patent #7205991*
16. Ravin Balakrishnan, Gordon Kurtenbach, George Fitzmaurice. (2006). System for physical rotation of volumetric display enclosures to facilitate viewing. *United States Patent #7138997*
17. Ravin Balakrishnan, Gordon Kurtenbach. (2005). A system for dynamically mapping input device movement as a user's viewpoint changes. *United States Patent # 6981229, 7,640,515*
18. Gordon Kurtenbach, George Fitzmaurice, Ravin Balakrishnan. (2004). Three dimensional volumetric display input and output configurations. *United States Patent #6753847, 7,583,252*
19. Ravin Balakrishnan, William Buxton, George W. Fitzmaurice, Gordon Kurtenbach. (2003). Digital tape drawing system. *United States Patent #6642927, 7,102,642*
20. Ravin Balakrishnan, Thomas Baudel, George W. Fitzmaurice, Gordon Kurtenbach. (2000). Three-dimensional input system using tilt. *United States Patent #6115028*.

Invited talks

Distinguished Lecture, University of New Brunswick, Faculty of Computer Science. Apr 2017.

Invited lecture, National University of Singapore. 26 Mar 2014. Host: Dr. Shengdong Zhao.

China HCI Symposium 2012. Invited distinguished Lecture. Chinese Academy of Sciences, Beijing, China. 18 June 2012. Hosts: Prof. Hongan Wang and Dr. Xiang Cao.

“Facile interaction with displays all over the place”. Invited lecture. Department of Computer Science, UOIT. Host: Dr. Faisal Qureshi. 26 Jan 2011.

“Facile interaction with displays all over the place”. Invited lecture. Department of Computer Science, Queens University. Host: Dr. Roel Vertegaal. 19 Jan 2010.

“Facile interaction with displays all over the place”. Colloquim. Department of Computer Science, Harvard University. Host: Dr. Chia Shen. 5 Mar 2009.

“Facile interaction with displays all over the place”. HP Labs India Lecture Series. Bangalore, India. 17 Feb 2009.

“Interacting with displays all over the place.” Invited talk at the UK-India Winter School on HCI. Bangalore, India. Host: Sriram Subramanian. 3 Feb 2009.

“Interacting with displays all over the place.” Invited talk at the McGill University CRMMRT Workshop on Multimodal Interaction. Host: Dr. Jeremy Cooperstock. 26 Sep 2008.

“Interacting with displays all over the place.” Invited talk at the iCORE Banff Informatics Summit. Host: Dr. Saul Greenberg and Dr. Sheelagh Cpendale. 22 Aug 2008.

“Towards more casual and realistic user interfaces.” Invited talk at HP Labs, Bangalore, India. Host: Dr. Sriganesh Madvanath. 20 Feb 2008.

“Towards more casual and realistic user interfaces.” Keynote talk at TECS Week, Pune, India. Host: Dr. Harrick Vin. 10 Jan 2008.

“Towards more casual and realistic user interfaces.” Keynote talk at Conference on Human-Machine Environments, Jinan, Shangdong Province, China. Host: Dr. Yuanchun Shi. 15 Oct 2008.

“Towards more casual ubiquitous interfaces.” Invited talk at Microsoft Research India, Bangalore. Host: Dr. Kentaro Toyama. 12 June 2007.

“Physics, projection, piles, and pens: user interfaces in the post GUI era.” Invited talk at Institute of Software, Chinese Academy of Sciences, Beijing. Host: Dr. Guozhong Dai. 5 Sep 2006.

“Physics, projection, piles, and pens: user interfaces in the post GUI era.” Invited talk at Tsinghua University, Beijing. Host: Dr. Yuanchun Shi. 1 Sept 2006.

“Physics, projection, piles, and pens: user interfaces in the post GUI era.” Invited talk at Microsoft Research Asia, Beijing. Host: Dr. Jian Wang. 30 Aug 2006.

“UI Projects at DGP lab”. Informal talk at Microsoft Research Cambridge. Host: Dr. Abigail Sellen. 12 July 2006.

“Physics, projection, piles, and pens: user interfaces in the post GUI era.” Invited talk at the University of Lille, France. Host: Prof. Gery Casiez. 10 July 2006.

“Physics, projection, piles, and pens: user interfaces in the post GUI era.” Invited talk at the iLog, France. Host: Dr. Thomas Baudel. 6 July 2006.

“Physics, projection, piles, and pens: user interfaces in the post GUI era.” Invited talk at LRI, University of Paris-Sud, France. Host: Prof. Michel Beaudouin-Lafon. 20 June 2006.

“Why aren’t we using 3D interfaces, and will we ever?” Keynote lecture, IEEE Symposium on 3D User Interfaces, 26 March 2006.

“Interaction and Visualization Techniques for Next-Generation Displays.” HCIL Colloquium, Department of Computer Science, University of Maryland, College Park. Host: Prof. Ben Bederson. 15 March 2005.

“Interaction Design for Future Computing Environments.” Colloquium, Department of Computer Science, University of Washington. Host: Prof. James Landay. 1 March 2005.

“Interaction and Visualization Techniques for Next-Generation Displays”. Distinguished Lecture, Department of Electrical Engineering and Computer Science, University of California, Berkeley. Host: Prof. John Canny. 17 Nov 2004.

“Usability challenges of large scale off-the-desktop interaction”. Invited talk at the IBM MITE Conference, Toronto, Canada. 4 Oct 2004.

“High degree-of-freedom interfaces”. Invited talk at the TTI/Vanguard Technology Transfer Symposium. Phoenix, Arizona: 3-4 Dec 2003.

“Non-wimp user interfaces”. Invited talk at IBM Centre for Advanced Studies. Host: Dr. Paul Smith. 26 Aug 2003.

“Fluid interaction techniques for the control and annotation of digital video”. Invited talk at Nokia Research Centre, Helsinki, Finland. Host: Miika Silfverberg. 9 May 2003.

“Interaction techniques for tabletop and volumetric displays”. Invited talk at Swedish National Defence College, Stockholm, Sweden. Host: S. Anders Christensson. 8 May 2003.

“Interaction with high degree-of-freedom input and next generation display technologies”. Invited lectures spanning 2 days at the University of Tampere, Tampere, Finland. Host: Prof. Kari-Jouko Räihä. 5-6 May 2003.

“High degree-of-freedom interaction and visualization”. Presentation at an invitation-only workshop at the Banff Centre for the Arts, Banff, Alberta. Host: Sara Diamond. 12-15 Sep 2002.

“Multi-channel and multi degree-of-freedom interaction for computer graphics”. Invited talk at National Research Council, Ottawa. Host: Dr. Janice Singer. 20 Feb 2002.

“Devices and Interaction Techniques for Interfaces to 3D Graphics”. Invited talk at University Telekom, Melaka, Malaysia. Host: Dr. Abu Hasan Ismail, Dean. Dec 1998.

“Multi Stream and Multi Degree of Freedom Input for Desktop 3D Interaction”. Invited talk at Microsoft Research. Host: Dr. Ken Hinckley. 15 Oct 1998.

“Overview of Alias|wavefront User Interface Research Group's research projects”. Presentation at Toron-Nihon Workshop organized by Dr. Paul Milgram at the University of Toronto. 19 Mar 1998.

Teaching

Graduate courses taught as a faculty member

CSC2604, “Topics in Human Centered and Interdisciplinary Computing”, Sep-Dec 2013.

A research seminar in human-centered computing. This year, the focus was on mobile and cyber-physical computing.

CSC2524, “Topics in Interactive Computing”, Jan - Apr 2013.

A research seminar in human-computer interaction. This year, the focus was on mobile computing.

CSC2524, “Topics in Interactive Computing”, Sep - Dec 2010.

A research seminar in human-computer interaction. This year, the focus was on next generation user interfaces.

CSC2524, “Topics in Interactive Computing”, Sep - Dec 2006.

A research seminar in human-computer interaction. This year, the focus was on advanced “off-the-desktop” interfaces.

CSC2524, “Topics in Interactive Computing”, Jan - Apr 2006.

A research seminar in human-computer interaction. This year, the focus was on interactive information visualization.

CSC2524, “Topics in Interactive Computing”, Sep - Dec 2004.

A research seminar in human-computer interaction. This year, the focus was on audio and gesture based user interfaces and interaction techniques.

CSC2514, “Human-Computer Interaction”, Sep - Dec 2004.

This is the same course as CSC428 (the second undergraduate course on Human-Computer Interaction). The graduate students taking the course were required to do a more substantial research paper.

CSC2524, “Topics in Interactive Computing”, Sep - Dec 2003

A research seminar in human-computer interaction. This year, the focus was on sketch and gesture based user interfaces and interaction techniques.

CSC2514, "Human-Computer Interaction", Sep - Dec 2003.

CSC2524, "Topics in Interactive Computing", Sep - Dec 2002

A research seminar in human-computer interaction. This year, the focus was large and tabletop display user interfaces and interaction techniques.

CSC2514, "Human-Computer Interaction", Sep - Dec 2002.

CSC2514, "Human-Computer Interaction", Sep - Dec 2001

Undergraduate courses taught as a faculty member

CSC490, "Capstone Design Project", Sep - Dec 2006.

CSC490, "Capstone Design Project", Sep 2005 - Apr 2006.

CSC490, "Capstone Design Project", Sep 2004 - Apr 2005.

4th year undergraduate course. This is a new course I proposed and developed last year that has students working on substantial projects spanning several areas of computer science.

CSC428, "Human-Computer Interaction", Sep - Dec 2004

4th year undergraduate course. This second Human-Computer Interaction course at Toronto builds on the department's first course in HCI, CSC318. While the focus in CSC318 is largely on the design process, this second course focuses more on the underlying models of human-computer interaction, rigorous formal evaluation techniques, and research frontiers

CSC490, "Capstone Design Project", Sep 2003 - Apr 2004.

CSC428, "Human-Computer Interaction", Sep - Dec 2003

CSC428, "Human-Computer Interaction", Sep - Dec 2002

CSC318, "The Design of Interactive Computational Media", Jan - Apr 2002

3rd year undergraduate course on methodologies for designing and evaluating user interfaces. The first course on Human-Computer Interaction at the University of Toronto.

CSC428, "Human-Computer Interaction", Sep - Dec 2001

Undergraduate courses taught while I was a graduate student

CSC258, "Computer Organization", May - Aug 2000.

2nd year undergraduate course on digital logic design, machine languages, and computer system organization fundamentals.

CSC258, "Computer Organization", Jan - Apr 2000.

CSC258, "Computer Organization", Sep - Dec 1999.

CSC108, "Introduction to Computer Programming", May - Aug 1999.

1st programming course for computer science undergraduates, taking an object-oriented approach using Java.

CSC258, "Computer Organization", Jan - Apr 1999.

CSC340, "Information Systems Analysis and Design", Sep - Dec 1998.

3rd year undergraduate course on theory, tools, and techniques for information systems analysis and design.

CSC258, "Computer Organization", May - Aug 1998.

CSC340, "Information Systems Analysis and Design", Sep - Dec 1997.

CSC104, "The How and Why of Computing", Jan - Apr 1997.

1st year undergraduate course in computing for non computer science students.

CSC258, "Computer Organization", May - Aug 1995.

Teaching assistantships while I was a graduate student

CSC428, "Human Computer Interaction", Sep - Dec 1996.

CSC228, "File Structures and Data Management", Sep - Dec 1995, Jan - Apr 1995.

CSC258, "Computer Organization", Jan - Apr 1996, Sep - Dec 1994, May - Aug 1994, Jan - Apr 1994, Sep - Dec 1993.

Academic advising

Current postdoctoral fellows

Priyank Chandra, 2019 – ongoing. Co-advised with Syed Ishtiaque Ahmed.

Current Ph.D. students

Jiannan Li. Sep 2016 – ongoing, Co-advised with Tovi Grossman

Laura Di Chen, Jan 2018 – ongoing, Co-advised with Tovi Grossman

Current masters students

(note: the Master's degree at Toronto is a research degree requiring a thesis or paper describing original research)

Former postdoctoral fellows

1. Fanny Chevalier, Jan 2012 – Nov 2013, co-advised with Karan Singh and Sara Diamond.
2. Thomas Pietrzak, Jan 2011 – Sep 2011, sole advisor
3. William Soukoreff, May 2010 – Nov 2011, sole advisor
4. Seok-Hyung Bae, Sept 2007 – Feb 2010, co-advised with Karan Singh.
5. Tomer Moscovich, Feb 2007 – June 2008.
6. Jeremy Birnholtz, Sep 2005 – July 2007, co-advised with Ron Baecker, Saul Greenberg, Carl Gutwin.
7. Géry Casiez, Feb - Aug 2005
8. Pierre Dragicevic, Feb 2006 – Apr 2007.

Graduated Ph.D. students

1. Aakar Gupta, Graduated 2018. Sole advisor.
Thesis: "Extended Hand Attributes for Touch Input, Touch Output and Touchless Interaction"
2. Sajad Shirali-Shahreza. Graduated 2017. Co-advised with Yashar Ganjali.
Thesis: "FleXight: Flexible Information Channel for Software-Defined Networking"
3. Rorik Henrikson, Graduated 2016. sole advisor.
Thesis: "Facilitating Real-Time Sketch-Based Storyboards for Stereoscopic and Virtual Reality Environments"
4. Dustin Freeman. Graduated 2015, sole advisor
Thesis: "Improv Remix: A system for Improv Theatre Performers to record and re-project Stage Video using wholebody interaction"
5. Jian Zhao. Graduated Apr 2015, sole advisor
Thesis: "Interactive visual data exploration: a multi-focus approach"

6. Xiaojun Bi. Graduated June 2011, sole advisor
Thesis: "Processing desktop work on a large high-resolution display: studies and designs"
7. Daniel Vogel. Graduated April 2010, sole advisor.
Thesis: "Direct pen input and hand occlusion". Received award for best Canadian thesis in HCI for 2011.
8. Clifton Forlines. Graduated Nov 2008, sole advisor
Thesis: "User interfaces for visual search"
9. Abhishek Ranjan. Graduated Oct 2008, sole advisor
Thesis: "Automatic camera control for capturing collaborative meetings"
10. Xiang Cao. Graduated Oct 2008, sole advisor
Thesis: "Handheld projector interaction"
11. Tovi Grossman. Graduated Apr 2008, sole advisor
Thesis: "Interaction with volumetric displays"
12. Daniel Wigdor. Graduated Apr 2008, sole advisor
Thesis: "The design of table-centric interactive spaces"
13. Gonzalo Ramos. Graduated Jan 2008, sole advisor
Thesis: "Pressure sensitive pen interactions"
14. Anastasia Bezerianos. Graduated Sept 2007, sole advisor
Thesis: "Designs for single user, up-close interaction with wall-sized displays"
15. Michael McGuffin. Graduated May 2007, sole advisor
Thesis: "An investigation of issues and techniques in highly interactive computational visualization"

Graduated masters students

(note: prior to 2007, the master's degree at Toronto required a formal thesis; since 2007, the thesis is optional and a research paper can be submitted instead).

1. Laura Di Chen. Graduated Jan 2018, sole advisor.
Research paper: "Learning by Interacting: an Investigation of Multi-Modal Communication in Live Streaming for Language Learning".
2. Rinat Abdrashitov. Graduated Jan 2015. co-advised with Karan Singh.
Research paper: "Interactive Shape Modeling using a Skeleton-Mesh Co-Representation".
3. Jonathan Deber. Graduated Jan 2013, sole advisor
Research paper: "Métamorphe: augmenting hotkey usage with actuated keys".
4. Stephanie Santosa. Graduated Jan 2013. co-advised with Karan Singh
Research paper: "Direct space-time trajectory control for visual media editing".
5. Aakar Gupta. Graduated Jan 2012, sole advisor
Research paper: "mClerk: Enabling Mobile Crowdsourcing in Developing Regions".
6. Rorik Henrikson. Graduated Jan 2012, sole advisor
Research paper: "An exploration of multi-user games using projector phones".
7. Navkar Samdaria. Graduated Jan 2012, sole advisor
Research paper: "Paying in kind for crowdsourced work in developing regions".
8. Mohit Jain. Graduated Jan 2011, sole advisor
Research paper: "An exploratory study of mobile collaborative learning in developing regions".
9. Akhil Mathur. Graduated Jan 2011, sole advisor
Research paper: "An exploratory study of a cameraphone-projector system".
10. Dustin Freeman. Graduated Sep 2010, sole advisor.

- Research paper: “Tangible Actions: More efficient direct multitouch through replaying gestures”.
11. Dominika Ruszkowski. Graduated Sep 2010, sole advisor.
Research paper: “Exploring user adaptability of mouse and touch in the desktop environment”.
 12. Patrick Dubroy. Graduated May 2009, sole advisor
Research paper: “A field study on the use of tabs in Mozilla Firefox”.
 13. Azam Khan. Graduated Sep 2008, sole advisor
Research paper: “ViewCube: A 3D orientation indicator and controller”.
 14. Michael Jurka. Graduated Jan 2008, sole advisor
Research paper: “Visualizing user activity to enhance document retrieval”.
 15. Gerald Chu. Graduated Dec 2007, sole advisor
Research paper: “Haptic conviction widgets”
 16. Tristan Campbell. Graduated Oct 2006, sole advisor
Thesis: “MCurve: A crossing based system for graphical music composition”
 17. Steven Tsang. Graduated Aug 2006, co-advised with Karan Singh.
Thesis: “A sketch based 2D animation system using dynamic motion hierarchies”
 18. Anand Agarawala. Graduated Jan 2006, sole advisor
Thesis: “Pushing the desktop metaphor with physics, piles and the pen”
 19. Daniel Vogel. Graduated Jan 2005, sole advisor
Thesis: “Interactive public ambient displays”
 20. Jia Sheng, Graduated Jan 2005, co-advised with Karan Singh.
Thesis: “A gestural 3D modeling interface using fingers and a physical prop tracked in 3D”
 21. Abhishek Ranjan. Graduated Jan 2005, sole advisor
Thesis: “Browsing archived meeting audio and time-synchronized data”
 22. Wael Aboelsaadat. Graduated Jan 2004, sole advisor
Thesis: “Evaluation of a transparent two-layer display”
 23. Xiang Cao. Graduated Jan 2004. co-advised with Allan Jepson.
Thesis: “An exploration of gesture-based interaction”
 24. Tovi Grossman. Graduated Jan 2004, sole advisor
Thesis: “Alternate user interfaces for the manipulation of curves in 3D environments”
 25. Daniel Wigdor. Graduated Jan 2004, sole advisor
Thesis: “Chording and tilting for rapid, unambiguous text entry to mobile phones”
 26. Yan (Grace) Wang. Graduated Dec 2003, sole advisor
Thesis: “Selection techniques for laser pointer interaction”
 27. Michael McGuffin. Graduated Aug 2002, co-advised with Gordon Kurtenbach
Thesis: “Fitts’ law and expanding targets: an experimental study, and applications to user interface design”

Ph.D. thesis committees

Karin Nieuwenhuizen. 2015. Department of Industrial Design, Eindhoven University of Technology. PhD external examiner. Advisor: Martens Jean-Bernard

Abir Mohammed Al Hajri. 2014. Department of Electrical Engineering, University of British Columbia. PhD external examiner. Advisor: Sid Fels.

Michael Gluek. 2012- present. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Daniel Wigdor

Marc Fiume. 2012- 2014. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Michael Brudno

Carrie Demmans Epp. 2012- present. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Ron Baecker

Vincent Cheung. 2013. Department of Electrical and Computer Engineering, University of Toronto. Ph.D. Thesis committee member. Advisor: Brendan Frey.

Felipe Bacim, 2011-present. Department of Computer Science, Virginia Tech. Ph.D. Thesis committee member. Advisor: Doug Bowman.

Mathieu Nancel. 2012. University of Paris, Sud. Ph.D. Thesis external reader. Advisor: Michel Beaudoin-Lafon.

William Soukoreff. 2010. York University. Ph.D. Thesis External Examiner. Advisor: Scott MacKenzie.

Craig Tashman, 2009 – 2012. GVV Centre, Georgia Tech. Ph.D. Thesis committee member. Advisor: Keith Edwards.

Regis Kopper. 2008 - 2012. Department of Computer Science, Virginia Tech. Ph.D. Thesis committee member. Advisor: Doug Bowman.

Michael Daum. 2006 – 2011. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Karan Singh.

David Dearman. 2006 – 2011. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Khai Truong.

Bowen Hui. 2003 – 2011. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Craig Boutilier

Shahzad Malik. 2005 – 2007. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Allan Jepson.

Iqbal Mohamed. 2005 – 2008. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Eyal de Lara.

Tao Ni. 2006 - 2011. Department of Computer Science, Virginia Tech. Ph.D. Thesis committee member. Advisor: Doug Bowman.

Koji Yatani. 2006 – 2011. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Khai Truong

Shengdong Zhao. 2004 – 2008. Department of Computer Science, University of Toronto. Ph.D. Thesis committee member. Advisor: Mark Chignell.

Maciej Kalisiak. 2002 - 2006. Department of Computer Science. Thesis committee member. Advisor: Michiel van de Panne

Wenbi Wang. 2004. Department of Mechanical and Industrial Engineering. Thesis defense committee member. Advisor: Paul Milgram.

Second reader for masters theses advised by others

Peter Hamilton. 2013. Advisor: Daniel Wigdor

Michael Glueck. 2012. Advisor: Daniel Wigdor

Frank Li. 2011. Advisor: Khai Truong

Yannick Thiel. 2011. Advisor: Karan Singh

James McCrae. 2008. Advisor: Karan Singh

Noah Lockwood. 2006. Advisor: Karan Singh

Naiqi Weng. 2005. Advisor: Karan Singh
Krista Strickland. 2005. Advisor: Ron Baecker
Xia Liu. 2004. Advisor: Karan Singh
Iqbal Mohamed. 2004. Advisor: Eyal de Lara
Michael Tsang. 2002. Advisor: Bill Buxton & George Fitzmaurice
Nicolas Burtnyk. 2002. Advisor: Bill Buxton

Undergraduate research students

Muhammad Anwar. Sep 1 2015 – Apr 30 2016. Engineering Science thesis.
Phoebe Chang. 1 May – 31 Aug 2013. (NSERC USRA award).
Curren Pangler, 1 May – 31 Aug 2007 (NSERC USRA award), co-advised with my postdoc Tomer Moscovich.
Sameer Vohra, 1 May – 31 Aug 2007 (NSERC USRA award), co-advised with my postdoc Tomer Moscovich.
Nicholas Kong, 1 May - 31 Aug 2005, 1 May - 31 Aug 2006 (NSERC USRA Award), 1 May – 31 Aug 2007 (NSERC USRA award), co-advised with my PhD student Tovi Grossman.
Matthew Boulos, 1 May - 31 Aug 2003 (NSERC USRA award) and 1 May - 31 Aug 2004 (NSERC USRA award), co-advised with my PhD student Gonzalo Ramos
Vikas Jain, Intern from IIT Bombay, 1 May - 31 July 2003, sole advisor
Tovi Grossman, 1 May - 31 Aug 2002 (NSERC USRA award), sole advisor
Jeffrey Magder, 1 May - 31 Aug 2002 (NSERC USRA award), sole advisor
Abhishek Ranjan, Intern from IIT Bombay, 1 May - 31 July 2002, sole advisor
Plus roughly 20 other undergraduates informally advised on various research activities in my lab.

Undergraduate project course supervision

(these are undergraduate research projects for which the student gets a CSC494/495 course credit or an Engineering Thesis)

Elaine Wong. Sep 2009 – Apr 2010.
Janek Biczysko-Lasocki, Jan – Apr 2007, co-advised with my PhD student Xiang Cao
Jacky Jie Li, Jan – Apr 2007, co-advised with my PhD student Xiang Cao
Phillip Mach, Jan – Apr 2007, co-advised with my PhD student Abhishek Ranjan
Clarence Chan, Jan - Apr 2004, co-advised with my PhD student Daniel Wigdor.
Steve Tsang, Sep - Dec 2002, sole advisor
Jeffrey Magder, Sep - Dec 2002, sole advisor
Daniel Wigdor, Jan - Apr 2002, sole advisor
Simone Maillard, Jan - Apr 2002, sole advisor
Tovi Grossman, Jan - Apr 2002, sole advisor
Wai Shan Lau, Jan - Apr 2002, sole advisor
Casson Ming Tse Lee, Jan - Apr 2002, sole advisor
Tovi Grossman, May - Aug 2000, co-advised with Gordon Kurtenbach.

Service outside the university

Journal, conference and workshop chairing and organization

ACM Transactions on Computer-Human Interaction. Associate Editor, 2010 – present.
Springer Computer Science HCI Series, Editorial Board, 2013-present.
Mobile HCI 2012, Doctoral Consortium Chair.
IEEE Transactions on Visualization and Computer Graphics. Associate Editor, 2008 – 2011.
ACM UIST 2009 Symposium on User Interface Software and Technology, Sponsorship Chair.
IEEE 3DUI 2009 Symposium on 3D User Interfaces. Conference Co-Chair.
ACM UIST 2008 Symposium on User Interface Software and Technology, Sponsorship Co-Chair.
Human-Computer Interaction journal. Co-editor of Special Issue on Ubiquitous Multi Display Environments. 2007-2008.
ACM UIST 2007 Symposium on User Interface Software and Technology, Program Chair.
ACM UIST 2007 Symposium on User Interface Software and Technology, Sponsorship Co-Chair.
Ubicomp 2005 Workshop on "The Spaces in-between: Seamful vs. Seamless Interactions". Co-organizer.
Graphics Interface 2004 Conference, Conference and Program Co-Chair
Graphics Interface 2002 Conference, Posters and Demonstrations Chair
International Input Research Workshop, Ribeaucourt, France, Apr 2000. Organizing Committee.

Conference and workshop program committees

ACM SIGGRAPH Conference on Computer Graphics and Interactive Techniques (2010).
ACM CHI Conference on Human Factors in Computing Systems, Associate Chair for Papers (2002, 2003, 2004, 2005, 2006, 2009, 2010, 2011, 2013, 2015).
ACM CHI Conference on Human Factors in Computing Systems, Short Papers Program Committee (2001)
International Projector-Camera Systems Workshop (PROCAMS), Program Committee (2008).
ACM SIGGRAPH International Conference on Virtual-Reality Continuum and its Applications in Industry (VRCAI), Program Committee (2008).
ACM UIST Symposium on User Interface Software and Technology, Papers Committee (2003, 2004, 2005, 2006, 2010, 2015). Doctoral symposium committee (2010).
Graphics Interface Conference, Papers Program Committee (2002).
ACM VRST Symposium on Virtual Reality Software and Technology, Full Papers Program Committee (2003).
IUI International Conference on Intelligent User Interfaces, Workshop on Intelligent User Interfaces for Developing Regions, Program Committee (2008).
IEEE InfoVis Symposium on Information Visualization, Papers Program Committee (2006).
ACM I3D Symposium on Interactive 3D Graphics and Games, Papers Program Committee (2006).
NordiCHI Conference, Papers Program Committee (2004).
8th Eurographics Workshop on Virtual Environments, Barcelona, Spain, Papers Program Committee (2002).
5th International Projector-Camera Systems Workshop (PROCAM), Program Committee (2008).
HCI track for CISIS2009, International Conference on Complex, Intelligent and Software Intensive Systems, Fukuoka, Japan. Program Committee (2009).

ACM Dev Conference (2010).

Paper & manuscript refereeing

ACM CHI Conference on Human Factors in Computing Systems (1998, 1999, 2000, 2001, 2007, 2008, 2011, 2013)

ACM UIST Symposium on User Interface Software and Technology (2001, 2002, 2008, 2009, 2013)

ACM SIGGRAPH Conference on Computer Graphics and Interactive Techniques (2001, 2002, 2003, 2004)

ACM I3D Symposium on Interactive 3D Graphics (2001, 2003)

ACM Transactions on Computer Human Interaction (2002, 2004, 2005, 2007, 2008)

ACM Conference on Universal Usability (2000)

ACM Computing Surveys (2008)

Communications of the ACM (1999)

Eurographics Conference (2003, 2004, 2005)

Graphics Interface Conference (1998, 2003, 2005, 2010, 2011)

Human Computer Interaction journal (2005, 2006)

IBM Systems Journal (1997)

IEEE Computer (2004)

IEEE Transactions on Visualization and Computer Graphics (2004, 2005, 2006, 2007)

IEEE Computer Graphics and Applications (2005, 2006)

IEEE InfoVis Symposium on Information Visualization (2005)

Interact Conference (1999, 2000)

Information Visualization journal (2007)

International Journal of Human Computer Studies (2001, 2005, 2006, 2008)

Journal of New Music Research (2003)

MIT Press book review (2005, 2006)

Morgan Kaufmann Press book review (2006).

Ubicomp Conference (2006)

Wiley Press book review (2003)

Grant & fellowship reviewing

NSERC Discovery Grants review committee (2013-2015).

Ontario Ministry of Research and Innovation Grant review (2008)

Netherlands Organisation for Scientific Research, grant review (2008).

Austrian Erwin Schroedinger fellowship review (2008).

Swiss National Science Foundation grant reviews (2007)

NSERC New Media Grant review (2004)

NSERC Strategic Grant reviews (2002, 2003, 2004)

NSERC Discovery Grant reviews (2003, 2004, 2005, 2006)

CFI Grant review (2007)

External organizations, tenure and promotion reviews

External reviewer for one tenure case at the University of Lugano (2013).
External reviewer for INRIA's graphics, interaction and visualization research teams (2010).
External reviewer for one tenure case at the University of North Carolina, Charlotte (2010).
External reviewer for one full professor promotion case at York University. (2010)
External reviewer for one tenure case at Harvard University. (2009)
External reviewer for one appointment renewal case at the University of North Carolina, Charlotte. (2008)
External reviewer for one tenure case at the University of California, Berkeley. (2008)
External reviewer for one tenure case at Nanyang Technological University, Singapore. (2007).
External reviewer for one tenure case at the University of Paris-Sud. (2007)

Other activities

ACM UIST 2005 Session Chair
ACM CHI 2002 Conference Session Chair
ACM CHI 2001 Conference Session Chair
Member of Association for Computing Machinery (ACM)

Service within the University of Toronto

Research and Industrial Relations Committee chair, Department of Computer Science, 2013-2014 academic year.
Faculty recruiting committee, Department of Computer Science, 2013-2014 academic year.
Graduate Admissions Committee., Department of Computer Science, 2012-2013 academic year.
PTR Committee., Department of Computer Science, 2010-2011 academic year.
Professional Masters committee, Department of Computer Science, 2009-2010 academic year.
PTR Committee., Department of Computer Science, 2009-2010 academic year.
Committee developing a proposal for a new "EDU-C" unit at the university called the Information and Communications Forum. Led by Prof. Alberto Leon-Garcia. 2008-2009 academic year.
Faculty recruiting committee, Department of Computer Science, 2008-2009 academic year.
Professional Masters committee, Department of Computer Science, 2008-2009 academic year.
Tenure committee for one tenure case at the Department of Mechanical and Industrial Engineering, 2009.
Evaluation committee for one 3rd year faculty review case at the Department of Computer Science, 2008.
Tenure committee for one tenure case at the Department of Computer Science, 2007.
Teaching evaluation committee for one tenure case at the Department of Computer Science, 2007.
Faculty recruiting committee, Department of Computer Science, 2006-2007 academic year.
Initiated and ran a seminar series on research skills and related topics intended for all graduate students in the department. The seminars are archived at: <http://www.dgp.toronto.edu/~ravin/gradskillsseminars2007/>
Graduate admissions committee, Department of Computer Science, 2005-2006 academic year.
Faculty recruiting committee for a HCI position, Department of Computer Science, 2004-2005 academic year.
Online graduate applications system: I took the lead on developing this system for the Department of Computer Science, Sept 2004 – May 2006.

Graduate admissions committee, Department of Computer Science, 2004-2005 academic year.

Graduate admissions committee, Department of Computer Science, 2003-2004 academic year.

Graduate breadth evaluation committee, Department of Computer Science, 2003-2004 academic year.

Graduate breadth evaluation committee, Department of Computer Science, 2002-2003 academic year.

Undergraduate committee, Department of Computer Science, 2002-2003 academic year.

Faculty support staff committee, Department of Computer Science, 2002-2003 academic year.

Department computing committee, Department of Computer Science, 2002-2003 academic year.

Faculty recruiting committee, Department of Computer Science, 2002-2003 academic year.

Department of Computer Science “planning discussion paper” committee, responsible for putting together a discussion paper on the direction of the department over the next 5-10 years. Apr-May 2003.

Environment and equity committee, Department of Computer Science, 2001-2002 academic year.

Initiated and organized, together with Kyros Kutulakos, Karan Singh, and Sam Roweis, the first visit of potential graduate students to the department. Feb-Mar 2002. I was also involved in organizing the event the next year (2003).

Initiated and organized a new process for graduate student recruiting where we invite short-listed candidates to visit us for a combined interview/recruitment event, *before* admission offers are made. We have found this to be a very successful process for making more informed admissions decisions. First organized in 2003 for the HCI group, then expanded to Graphics and HCI in 2004, and since then other groups in the department have begun to adopt this approach as well.

From July 2001 to September 2006, I was responsible for the day-to-day direction and management (including supervision of 1 full-time and 1 part-time technical staff members) of the Dynamic Graphics Project laboratory (www.dgp.toronto.edu), which is home base to the department’s graphics and HCI faculty and graduate students (7 faculty, 50 students).

Graphics and HCI seminar coordinator, Department of Computer Science, 2001-2002, 2002-2003, 2003-2004, 2004-2005, 2005-2006 academic years.