UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

BOX, INC., and DROPBOX, INC.,

Petitioner

v.

TOPIA TECHNOLOGY, INC.,

Patent Owner

Case IPR2023-00431
Patent 10,642,787

Case IPR2023-00432
Patent 10,754,823

DECLARATION OF PRASHANT SHENOY, PH.D.

TABLE OF CONTENTS

I. BACKGROUND AND QUALIFICATIONS 3

II. LEGAL FRAMEWORK 6

A. Anticipation 6

B. Obviousness 7

III. OPINION 9

A. Claim Construction 10

B. Background of the Technology 10

C. '787 Ground 1: The Petition Has Failed to Establish Unpatentability for Challenged Claims Based on Sigurdsson, Shappell, and Rao 13

D. '787 Ground 2: The Petition Has Failed to Establish Unpatentability for Challenged Claims Based on Brown, Hesselink, Shappell, and Rao 28

E. '823 Ground 1: The Petition Has Failed to Establish Unpatentability for Challenged Claims Based on Sigurdsson, Shappell, and Rao 36

F. '823 Ground 2: The Petition Has Failed to Establish Unpatentability for Challenged Claims Based on Brown, Hesselink, Shappell, and Rao 50

IV. CONCLUSION 59

I. BACKGROUND AND QUALIFICATIONS

1. I am a Distinguished Professor and Associate Dean in the College of Information and Computer Sciences at University of Massachusetts, Amherst. I practice, research, and teach broadly in the area of Distributed Systems and Networking.

2. I hold a PhD. (1998) and an M.S. (1994) in Computer Science from The University of Texas at Austin, where I was awarded the best dissertation award in Computer Science. I also hold a B. Tech (1993) in Computer Sciences from the Indian Institute of Technology (IIT) Bombay, where I was awarded the IIT Bombay Silver Medal for the top ranking student in Computer Science.

3. I have published over 300 technical papers in reputed scientific journals and conferences, and those publications have been cited 24,000 times. My research has received several awards, including best paper awards and the ACM Sigmentrics Test of Time Award. I am currently a Fellow at four professional computer societies – Association for Computing Machinery (ACM), Institute of Electrical and Electronics Engineers (IEEE), American Association of Advancement of Science (AAAS), and Asia-Pacific Artificial Intelligence Association (AAIA).

4. I have over 25 years of practical, research, and teaching experience in distributed systems, file systems and management systems, data storage, backup, and networking. I have led research projects and industry collaborations in storage

and file systems, web and internet systems, content distribution, streaming media, database systems, cloud computing, virtualization, RFID and sensor networks, and energy systems. In my 25 years of experience, I have consulted with both large companies and startups in a technical advisory role related to product architecture, IP issues, and applied research. My qualifications are further summarized in my *curriculum vitae*, which is submitted as EX 2025.

5. I have been retained by Plaintiff Topia Technology, Inc., ("Topia") to provide my independent opinion on certain issues in the matter. I do not have any financial interest in Topia or the outcome of this matter. My compensation in this matter is independent of the substance of my opinions and analysis in this matter or in the outcome of this matter.

6. In connection with this matter, I have studied and reviewed the following materials:

EXHIBIT	DESCRIPTION
1001	U.S. Patent 9,143,561 ("'561 Patent")
1002	U.S. Patent 10,067,942 ("'942 Patent")
1003	U.S. Patent 10,289,607 ("'607 Patent")
1004	U.S. Patent 10,642,787 ("'787 Patent")
1005	U.S. Patent 10,754,823 ("'823 Patent")
1006	U.S. Patent 11,003,622 ("'622 Patent")
1007	Prosecution History of the '561 Patent

1008	Prosecution History of the ’942 Patent
1009	Prosecution History of the ’607 Patent
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
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 Rajest M. Rao, et al. (“Rao”)
1020	U.S. Patent Application Publication No. 2005/0091289 A1 to Michael Shappell, et al. (“Shappell”)
1023	European Patent Application Publiciation 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 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")
Paper No. 3	Petition for IPR of the '561 Patent (IPR2023-00427)
Paper No. 3	Petition for IPR of the '942 Patent (IPR2023-00433)
Paper No. 3	Petition for IPR of the '607 Patent (IPR2023-00429)
Paper No. 3	Petition for IPR of the '787 Patent (IPR2023-00431)
Paper No. 3	Petition for IPR of the '823 Patent (IPR2023-00432)
Paper No. 3	Petition for IPR of the '622 Patent (IPR2023-00430)

II. LEGAL FRAMEWORK

7. I am a technical expert and do not offer legal opinions. However, I have been informed about certain legal principles regarding patentability and related matters under United States patent law, which I have applied in performing my analysis and arriving at my technical opinions in this matter.

A. Anticipation

8. I have been informed and understand that a patent claim is invalid as "anticipated" if each and every feature of the claim is found in a single prior art reference. I also understand that anticipation requires the presence in a single prior art a description of all elements of the claim arranged as required in the claim. I have been informed that for a reference to be considered as anticipating, it must disclose the relevant technology in a manner such that a person of ordinary skill in the art ("POSA") would be able to carry out or utilize the technology that the

reference describes without having to undertake undue experimentation.

B. Obviousness

9. I have been informed and understand that a patent claim is not patentable if the claimed invention would have been obvious to someone of ordinary skill in the art at the time the patent application was filed. This means that even if all of the requirements of the claim cannot be found in a single prior art reference so as to anticipate the claim, the claim might still not be patentable if the subject matter in the claim would have been obvious. I also have been informed that a claim would have been obvious when the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to someone of ordinary skill in the art.

10. I have been informed and understand that to prove that a prior art reference or a combination of prior art references renders a patent obvious, it is necessary to: (1) identify the particular references that either alone, or in combination, render the patent obvious; (2) specifically identify which elements of the patent claim appear in each of the asserted references; and (3) explain how the prior art references could have been combined in order to create the inventions defined in the particular claim at issue.

11. I have been informed that, in order to establish that a claimed invention

would have been obvious based on a combination of prior art documents, a clear articulation of the reason(s) why the claimed invention would have been obvious, or why a certain prior art document would have been modified, must be provided.

12. I have been informed and understand that for a patent claim to have been obvious, the claim must be obvious to someone of ordinary skill in the art at the time of the alleged invention. I have been informed that the factors to consider in determining the level of ordinary skill in the art include (1) the educational level and experience of people working in the field at the time the invention was made, (2) the types of problems faced in the art and the solutions found to those problems, and (3) the sophistication of the technology in the field.

13. I also have been informed and understand that it is improper to combine references where the references teach away from their combination. I have been informed that a prior art reference may be said to teach away when a person of ordinary skill in the art, upon reading the prior art reference, would have been led in a direction divergent from the path that was taken by the inventors.

14. I have been informed and understand that a final determination of obviousness should also consider secondary considerations, if any, such as commercial success of products covered by the patent. I understand that to date secondary considerations have not been raised in these proceedings.

15. It is my understanding that for purposes of this proceeding, Petitioners are

of the opinion that a POSA in November 2007 would have been: (1) a person with an undergraduate degree in computer science or closely related field and (2) two or more years of experience with systems and architecture for file management in a distributed network environment.

16. Based on my education, training, and professional experience in the field of the claimed invention, I exceed the level and abilities of a POSA at the time of the claimed invention. Although my qualifications may exceed those of one skilled in the art, my analysis and opinions regarding the Patents have been rendered from the perspective of one skilled in the art at the time of the invention.

III. OPINION

17. In my opinion, claims 1-4, 8-11, and 13-16 of the '787 Patent are patentable based on my review of the '787 Patent, the file history, and scope of the references discussed herein and in the petition, the knowledge and skill of a POSA would have has at the time of the invention and priority date of the '787 Patent, and the documents listed in the above-mentioned table.

18. In my opinion, claims 1-4, 8-11, and 13-16 of the '823 Patent are patentable based on my review of the '823 Patent, the file history, and scope of the references discussed herein and in the petition, the knowledge and skill of a POSA would have has at the time of the invention and priority date of the '823 Patent, and the documents listed in the above-mentioned table.

19. I have also relied on my considerable academic and practical experience about the relevant field of technology in forming my opinions as expressed herein.

A. Claim Construction

20. In my analysis I have interpreted the claims in accordance with their plain and ordinary meaning, as instructed. In my opinion, these interpretations are consistent with the claims, specification, and knowledge of a person of ordinary skill in the art.

B. Background of the Technology

21. The ’787 Patent[1], including the claims thereof, relate to pre-file transfer update based on prioritized metadata. EX1004, Title, Abstract.

22. The ‘787 Patent discloses a system to perform full-file synchronization across multiple local devices associated with a user through a server, e.g., a user centric architecture with transfer of prioritized metadata before the transfer of the file copy. EX1004, Title, Abstract.

23. The background of the ‘787 Patent describes various problems in connection with computerized file management systems circa 2007/2008 timeframe. *Id.*, 1:30-3:50. As the‘787, Patent explains, at least as late as 2007, users of modern computing systems were increasingly finding themselves in high-speed networked

[1] While U.S. Patent No. 10,642,787 (the “’787 patent”), and U.S. Patent No. 10,754,823 (the “’823 patent”) are discussed in this declaration, since these patents share a specification, references have been made to the specification of the ‘787 patent.

environments. *Id.*, 1:30-35. With the recent proliferation of personal computing devices, these users would have several computing devices, such as a desktop computer at home, one at work, and a handheld device, such as a personal digital assistant (PDA), on which they would access their device files. *Id.* A user having multiple computing devices led to problems with managing the user's files across those various devices. *Id.*, 1:38-68. Users, or customers, needed and desired a seamless environment for accessing their files across their various electronic devices without having to worry about the details of getting the file, and any of its related components, transferred between those devices. *Id.*

24. Although a user could copy files to have them accessible on the user's other devices, this often resulted in a proliferation of redundant file copies that created confusion about which was the latest or "official" version of the file. *Id.*, 1:45-50. This conventional process was tedious to resolve, error prone, and complex. *Id.*, 1:45-65.

25. Since copying of files across a user's devices was tedious and complex, another approach for accessing files from various devices was for users to store the file at a central location, such as a file server. *Id.*, 1:68-2:45. At that time there were several automated solutions to allow a user to access the most current version of the user's file from any of their devices, but these too had problems. *Id.* Remote desktop software, distributed file systems, and file transfer protocol (FTP)

systems purported to offer a way for a user to access the most current version of the user's files across their devices, but these methods relied on the files being stored in a single central location, such as the server, rather than locally on multiple devices of a user. *Id.* Because these solutions depended on the devices of the user being connected to a server through a network to access the user's file, the availability and performance of the network was a limiting factor for accessing and using the file. *Id.*, 2:24-30.

26. At the time, connections between a server and a user's local device (e.g., desktop computer, laptop, PDA, etc.) through a communication network, such as the Internet, often were intermittent and not as reliable as they are today. For a user to access the user's file residing on the server the user's device had to be connected to the server through the communication network. In this case, these computerized file management systems would store a document file on the server and not on the user's device. The user's device had to be connected through the network to the server to access and edit the document. If the network connection was down or inaccessible, the user would not be able to access the document file, or if the network was operating slowly, the user might not be able to edit the document file effectively because of the poor performance due to the slow communication connection.

27. Since server-centric systems stored files centrally on a server, alternative approaches were needed since the file became unavailable when the network was unavailable.

28. A solution that is discussed in the '787 Patent and '823 Patent is providing a user with an accurate "picture" of their document collection regardless of whether the user's devices physically contain the documents.

29. As discussed above, a problem with distributed file systems and FTP was the latency between a file being put onto a file system and it showing up on the user's machine. A solution to this problem discussed in the '787 Patent and '823 Patent is decoupling a directory from the movement of files by updating the directory system at a higher priority than the documents to be synchronized. *Id.*, 9:17-35. This ensures that when a user browses or searches through his set of documents, they appear even if they are not yet available on the user device.

C. '787 Ground 1: The Petition Has Failed to Establish Unpatentability for *Challenged Claims* Based on Sigurdsson, Shappell, and Rao

30. I have reviewed the Sigurdsson reference and in my opinion, the Sigurdsson reference teaches an HTTP server architecture which is fundamentally different from the alleged suggestions that Petitioners are citing it for.

31. According to the claims of the '787 Patent, a server system is caused to: 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 (e.g., claims 1, 8 and 13)).

32. Sigurdsson does not teach or suggest that "**the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file**", because the "priority algorithm" in Sigurdsson is for prioritizing between the content that is to be synchronized and not for prioritizing between metadata and the content. For example, in ¶58, Sigurdsson discloses three data items include 1) very large document such as a maintenance manual, 2) data from digital camera, and 3) a small office memo. Moreover, Sigurdsson discloses that, 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. As such, the server 106 can handle the higher priority data items first, and save the lower priority data items for later.

33. In FIG. 6, Sigurdsson discloses that the server receives a request for a list of missing content from a client (Client C) (612) and passes a missing content list to the client (Client C) (616) in response to an express request for a list of missing content from the client. ¶¶ 69-73. As such, the list of missing content is only provided in response to a request from the client, but there is no assignment of priority by the server. Moreover, this sequence of passing the "list of missing

content" from the server 106 to Client C in step 616 in response to Client C's request for missing content in step 612 disclosed in Sigurdsson also includes step 614 in which server 106 "determines the content missing from the third client" at the time Client C requests the list of missing content. Sigurdsson, FIG. 6, ¶70. Therefore, metadata received from the first device cannot have been assigned a priority either before or upon its receipt at the server, since Sigurdsson's so-called priority is assigned only when Client C submits a request to the server for missing content.

34. According to the claims of the '787 Patent, the server system is further caused to: **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. As such, it is the server system that "automatically transfers" the first metadata over a network to a second client device associated with the user. Sigurdsson does not teach anything but a server automatically transferring the first metadata.

35. To the contrary, Sigurdsson discloses a system, in which, a receiving device (i.e., a Client C 406) requests for a list of missing content. Only in response to the server receiving the request for the list from Client C does it transmit that list of missing content to Client C. (FIG. 6, ¶¶69-73). Therefore, in Sigurdsson, the server does not automatically transfer the first metadata to the receiving device,

such as Client C, since it only transfers the list of missing content when the receiving device expressly requests a list.

36. Not only does Sigurdsson fail to teach or suggest that a server **automatically transfers** the first metadata to the second client device, it also fails to teach that such a transfer is "**based on the first priority being greater than the second priority**." Sigurdsson, in FIG. 6 discloses a sequence diagram of a request/response based mixed peer-to-peer and client/server architecture, including a server, and clients A, B, and C. In FIG. 6, the server in Sigurdsson provides a **list** of missing content only in response to a request for such a list from a client (e.g., "Client C" at steps 612 and 616).

37. That is, in FIG. 6, client/server structure and operation of Sigurdsson requires an explicit request for a list of missing content from the client device ("Client C") to the server, before the server provides the list of missing content to the client device, which is neither an "**automatically transfer**" by the server nor "**based on the first priority being greater than the second priority**".

38. Sigurdsson, in ¶¶69-73 requires the following steps in order: (1) "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 …"; (2) in response, "[i]n step 614, the server determines the content missing from the third client …"; and subsequently, "[i]n step 616, the server passes the list

of missing content to the third client (e.g., Client C), which had been offline." It is only after going through this sequence of Client C expressly requesting the server for list of any missing content, and the server determining the content missing from the third client, the server responds by passing the list of missing content to Client C. As such, Sigurdsson does not teach or suggest that the server 106 **automatically transfers** the list of missing content (allegedly including the first metadata) to the Client C 406 "**based on the first priority being greater than the second priority**".

39. Since Sigurdsson discloses that the server only transfers the list of missing content *when requested* by the second client device, the transfer of the list of missing content to the second client device by the server in Sigurdsson is based on a request from the second client device and not automatically based on the first priority being greater than the second priority.

40. In FIG. 3, Sigurdsson discloses that the client devices 102 and 104 are implemented as a hypertext transfer protocol secure (HTTPS) client/server structure (*i.e.*, HTTPS Client/Server 256 and HTTPS Client/Server 356).

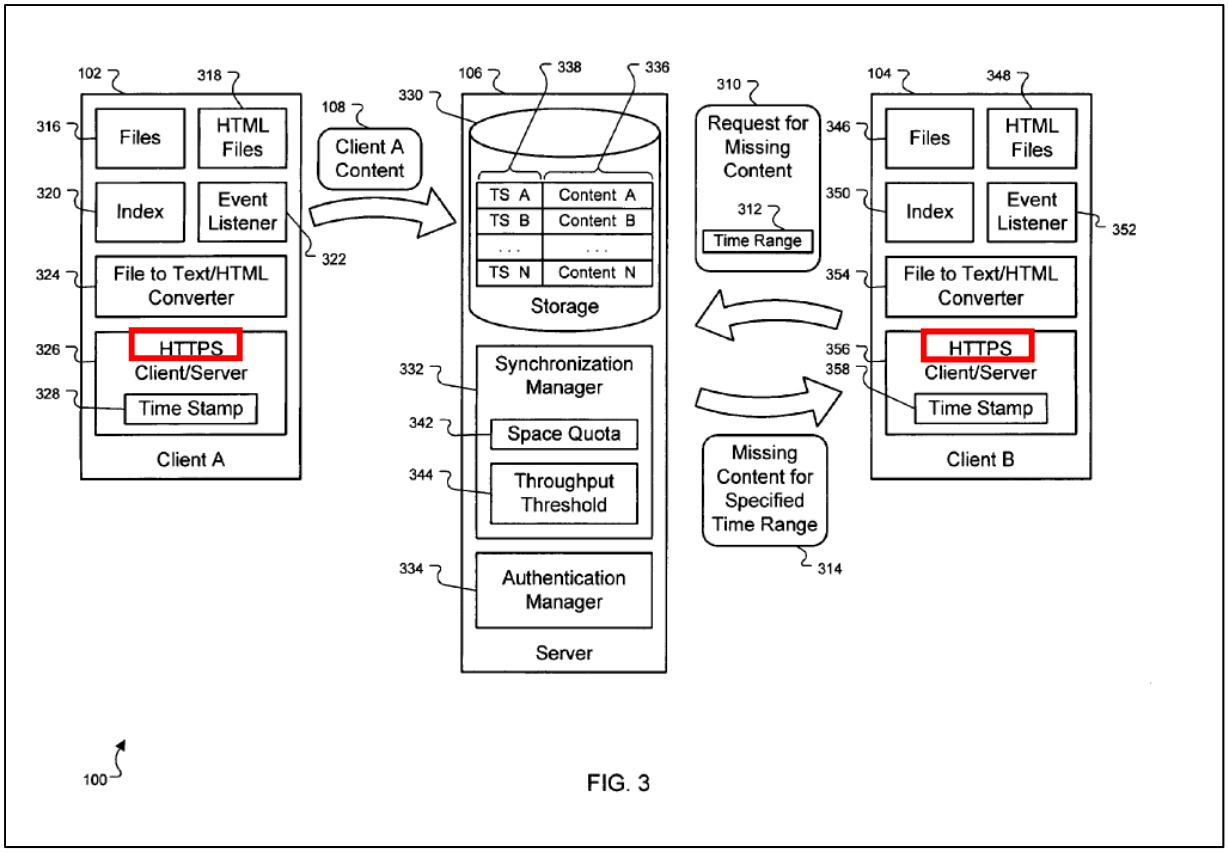


FIG. 3

Sigurdsson (EX 1016), FIG. 3

41. HTTP (Hypertext Transfer Protocol) is a protocol used to carry requests from a browser to a Web server and to transport pages from Web servers back to the requesting browser. (Microsoft Computer Dictionary, p.259), Moreover, a HTTP server is server software that uses the HTTP protocol to serve up HTML documents and any associated files and scripts when requested by a client, such as a Web browser (Microsoft Computer Dictionary, p. 260). An HTTP/HTTPS server (e.g., the web server) plays the role of a server in a client–server model by implementing the server part of the HTTP/HTTPS network protocol. An important feature of an HTTP/HTTPS server is that the HTTP/HTTPS server waits for

incoming client requests and for each request the HTTP/HTTPS server responds by providing the requested information or performing a requested task. The HTTP/HTTPS server does not perform actions on its own without a request from a client. Since the HTTP protocol is a request-response protocol, where clients send HTTP requests to the server and the server sends HTTP responses after processing each request, the HTTP protocol does not support a functionality where the server can automatically send data to a client without the client making requests for data.

42. FIG. 3 of Sigurdsson teaches a HTTPS server, which does not perform functions "automatically" without client requests for content. Thus, Sigurdsson requires a HTTPS server system based on an HTTP request/response process for data transfer between the client and the server as illustrated in FIG. 3. Therefore, Sigurdsson not only fails to teach the features of "**automatically transfer[ring]**, based on the first priority being greater than the second priority, **the first metadata** … to a second client device associated with the user," it would not have been obvious to modify the server system based on an HTTP request/response process of Sigurdsson to the automatically transfer the first metadata without any client request.

43. According to claim 3 (and similarly claims 10 and 15) of the '787 Patent, the computer program instructions, when executed, further **cause the server 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 … **receiving the copy of the second file from the second client device**.

44. Sigurdsson discloses a system, in which, a receiving device (i.e., a Client C 406) performs several operations before requesting a copy of missing content. Much of Sigurdsson discusses client devices requesting such missing content from other client devices, in a peer-to-peer arrangement. See Abstract. Sigurdsson describes an embodiment in which Client C obtains missing content from a server.

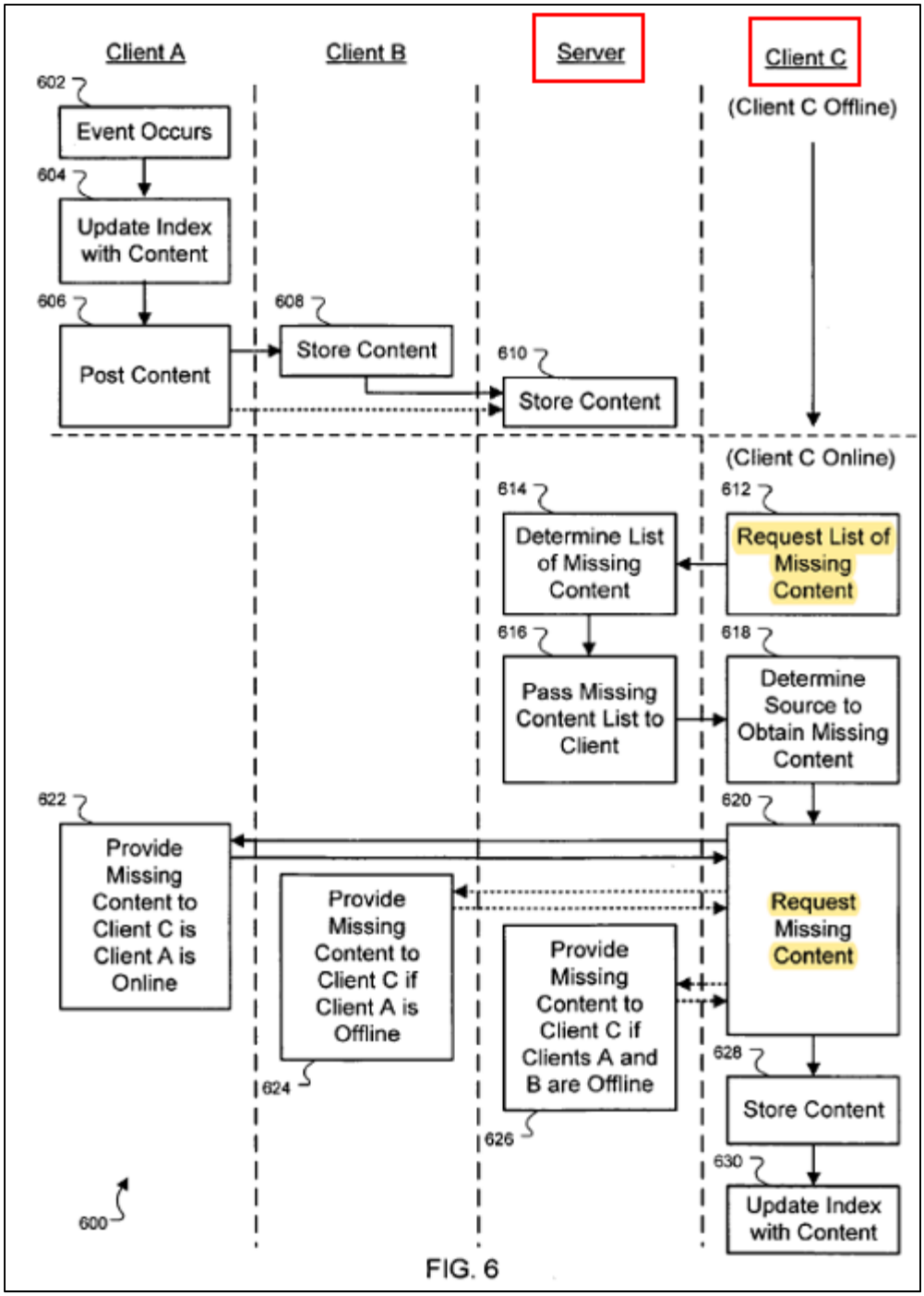
45. But that embodiment requires Client C to expressly request the missing content. Only in response to the server receiving the request from Client C does it transmit that content to Client C. (FIG. 4, 6, ¶¶71-75, 80). Therefore, in Sigurdsson, the server does not automatically transfer a modified copy of the file to the receiving device, such as Client C, since it only transfers the missing content when the receiving device expressly requests it.

46. The automatic transfer in the claims of the '787 Patent refers to the server system automatically transferring the copy of the first file.

47. Not only does Sigurdsson fail to teach or suggest that a server "**automatically transfer[s]** the copy of the second file to the first client device associated with the user ", it also fails to teach that such a transfer is "**responsive**

to … **receiving the copy of the second file from the second client device**." Sigurdsson's transfer to Client C is the result of Client C requesting the transfer.

48. Sigurdsson, in FIG. 6 (reproduced below with annotations), discloses a sequence diagram of a request/response based mixed peer-to-peer and client/server architecture, including a server, and clients A, B, and C. As shown in FIG. 6, the server in Sigurdsson provides missing content only in response to a request from a client (e.g., "Client C" at steps 612 and 620). That is, in FIG. 6, client/server structure and operation of Sigurdsson requires at least two separate requests from the client device ("Client C") to the server, before the server provides the missing content to the client device, neither of which are "receiving the copy of the first file from the first client device".



Sigurdsson (EX 1016), FIG. 6

49. It is only after going through this sequence of Client C requesting the server for any missing content, Client C determining where it can obtain that missing

content, and then requesting the content from the server, before the server responds by transferring the content to Client C. As such, Sigurdsson does not teach or suggest that the server 106 automatically transfers the content 410 to the Client C 406 responsive to receiving the content 410 from Client A 402.

50. Sigurdsson at FIG. 2 and ¶¶ 37, 40 and 46 discloses a request for missing content from the second client. For instance, "**the second client polls** for updates" (¶37), the request is "**periodically issue[d]", and "the request 310 includes a Time Range 312 parameter** identifying the range of time stamps for the missing content" ¶40 and "the Request 310 is issued **in order** for the Client B 104 to obtain the Client A Content 108 that the Server 106 has yet to send to the Client B104" (¶46).

51. FIG. 5 and the related text of Sigurdsson does not disclose several of the claimed features including the "**automatically transfer** the copy of the second file to the first client device", "responsive to … **(ii) receiving the copy of the second file from the second client device**" as discussed above." In fact, the disclosure in FIG. 5 simply summarizes the request-based file transfer approach from FIG. 6 discussed above in detail.

52. As discussed above, FIG. 3 of Sigurdsson teaches a HTTPS server, which does not perform functions "automatically" without client requests for content. Thus, Sigurdsson requires a HTTPS server system based on an HTTP

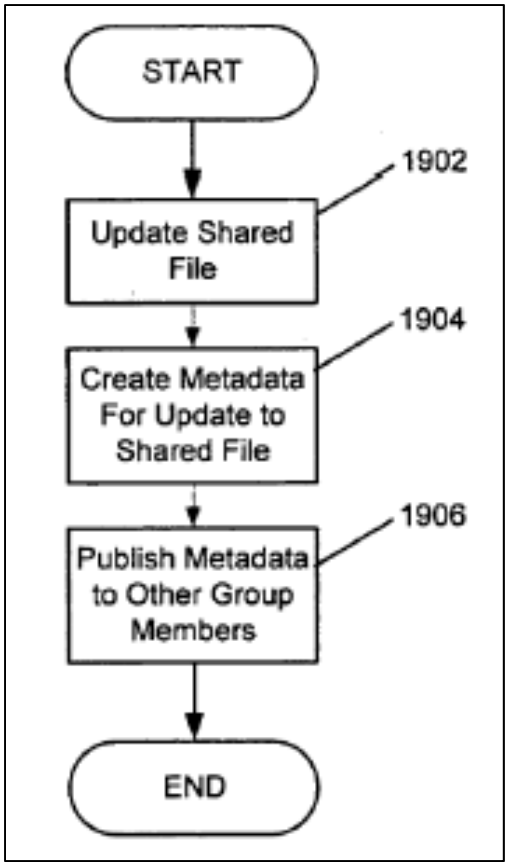
request/response process for data transfer between the client and the server as illustrated in FIG. 3. Therefore, Sigurdsson not only fails to satisfy the features of "**automatically transfer** the copy of the second file to the first client device", it would not have been obvious to replace the principal structure (i.e., the server system based on an HTTP request/response process) taught by Sigurdsson with the automatically transfer of a copy of the file to a client without any client request to disclose the claim features.

53. I have reviewed the secondary reference Shappell and in my opinion it discloses a peer-to-peer network environment which is fundamentally different from a Client-Server environment. ¶ 7 of Shappell. The Abstract of Shappell indicates that the disclosed system and methods disclosed are used to enable users to share files in a server-less shared space. By emphasizing a "server-less" environment, Shappell does not teach a Client-Server system that uses a server.

54. Shappell discloses "a computer implemented method and system enable users to share files in a **server-less** shared space. By providing access to such **server-less** spaces via a visual presentation, the system renders content available for access by other group members. Access is sometimes provided through propagation of metadata or other uniquely identifying indicia associated with the shared space to all group members." Abstract. Shappell teaches away from using a server, and does not teach or suggest a server system **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 receiving the copy of the first file from the first client device**.

55. Shappell is silent about a priority being assigned to the first metadata or the copy of the first file. In particular, in cited FIG. 19 (reproduced below), Shappell at most discloses "creating metadata for update to shared file" and "publishing metadata to other group members." However, there is no mention of a priority being assigned to the metadata. No other teaching in Shappell discloses a priority being assigned to the metadata. Therefore, Shappell does not teach or suggest "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file".



Shappell (EX 1020), FIG. 19

56. In FIG. 19, step 1906, Shappell discloses the metadata to all other group members, which is different from the claimed features of automatically transferring the first metadata **to the second client device associated with the same user**.

57. I have reviewed the secondary reference Rao and, in my opinion, it discloses the synchronization of the metadata and raw data in databases in a manner which is fundamentally different from that for which the Petition cites it. Paragraph 0002 of Rao.

58. Rao discloses the two-phase synchronization of a database, whereby, in the first phase, metadata is synchronized between a server and a client while raw data

remains on the server and, in the second phase, selected raw data is synchronized to the client. Paragraph 0063 of Rao.

59. Rao does not teach or suggest that "**the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file**" or the **"automatic[] transfer, based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device"**, because Rao instead focuses on the wholesale synchronization of a database in phases – first metadata transfer and then raw data transfer. Rao does not disclose the assignment of a greater priority to the metadata than to the file, and Rao instead transfers metadata before selected raw data irrespective of any such priority. Rather, Rao categorically transfers metadata before its associated raw data.

60. Furthermore, Rao is not in the same field as that of the '787 Patent and neither it is pertinent to the problem addressed by the '787 Patent, because it focuses on synchronizing raw data in a database rather than on synchronizing files between clients. Due to its focus on raw data in a database, a POSA would not have used Rao alone or in combination with other references with regards to the above features, "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file" or the "automatic[] transfer,

based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device".

D. '787 Ground 2: The Petition Has Failed to Establish Unpatentability for Challenged Claims Based on Brown, Hesselink, Shappell, and Rao

61. The Petition argues that Brown describes that the originating "first client device" sends to the server "CSI" (Brown, 10:9-10) and "parent directory SID," and "file name," *id*., 15:38-40, 5:50-52), which allegedly corresponds to metadata for a modified file, prior to the modified file being uploaded, citing *id.*, 13:57-62 and Fig. 8A (step 805), 14:5-6 and Fig. 8B (step 850), Figs. 11A-C and 15:30-54 (step 1115 and step 1120). Pet., 53-54.

62. However, the client sync index ("CSI") in Brown merely refers to a value indicating whether the client directory is up to date ("If the SSI value matches the CSI value passed in by the polling client, the client is up to date with the current state of its server account."; Brown, 5:17-19). The CSI received from a client is not "associated with the updated version" of the file that is generated from the user modifying the content of the file, as recited in claims [1c], [8b], and [13b]. Also, in Brown, the CSI is included in the sync poll call from the client Synchronization Application (SA) to the server, and the server returns its server sync index (SSI) back to the client in response. That is, the CSI as taught in Brown is merely a sync

call requesting the server's SSI so that the client can decide whether the CSI matches with the SSI (that is, whether the client directory is up to date).

63. Petition also relies upon step 1115 of FIG. 11A of Brown, which discloses that the client sends the SID, the parent directory SID, and the file name to the server, and argues that this portion teaches "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file" in claim [1c]. However, the SID, the parent directory SID, and the file name are transmitted to the server only for the purpose of locating, if any, a previous version of the file is on the server, at step 1117 in FIG. 11B. If there is no previous version of the file on the server, then at step 1120, *the file and the client metadata item* are uploaded to the server.

64. The Petition, with regards to claim [1d], cites to the server synchronization data ("SSD") in Brown as the metadata that the server first transfers to the receiving client before the server transfers a copy of the modified file to that receiving client. Pet., 54. However, the SSD contains server metadata items for any items with file sync index (FSIs) greater than the CSI passed in by the receiving client's sync poll call. Brown, 8:53-61. That is, the SSD can be generated *only after the CSI is received* from the receiving client's sync poll call. This is also consistent with Brown's disclosure that the SSD is generated responsive to a client sync index (CSI) received from the client. Brown, claims 9-

10. Further, because the SSD needs to include server metadata items for any items that are updated from the last sync call (that is, any items with FSIs greater than the client's current CS), Brown necessarily requires the client metadata received at step 1120 in FIG. 111B in order to generate the SSD. Therefore, the "SSD" of Brown does not teach "the first metadata that is *automatically* transferred to a second client device *based on the first priority being greater than the second priority*."

65. In Brown, the client SA needs to poll the server to check if there are any changes in the client's directory, and if the client's directory is not up to date, the client then requests server synchronization data (SSD) to learn about the new changes that have been made at server 105. ("If the CSI matches the server account's SSI value, then the client is up to date with the server. Otherwise *the client SA requests server synchronization data (SSD)*."; Brown, 8:51-53). If 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." Brown, 10:19-21. Based on the SSD received from the server, "the *client SA updates the client's directory and metadata* to match the server state." Brown, 8:62-63. That is, in Brown, the SSD are transmitted to the client device *upon the request received from the client*. *See e.g.,* Brown, 10:18-27

(the server sends the SSD to client 130 *in response to a request for the SSD by the client*).

66. Brown teaches a HTTP server, which uses a client-pull architecture and does not perform functions "automatically" without client requests. Brown, 4:12-15 ("[C]ommunication between the clients and server 105 is tunneled within the hypertext transport protocol (HTTP).")

67. As discussed above, the HTTP protocol is a request-response protocol, and the HTTP servers use the client-pull architecture, where clients send HTTP requests to the server and the server sends HTTP responses after processing each request. The HTTP protocol does not support push-based functionality where the server can automatically send data to a client without the client making requests for data.

68. Because Brown requires a HTTP server system based on an HTTP request/response process for synchronization, Brown fails to disclose the features related to automatically transferring the first metadata.

69. It would not have been obvious to replace the pull-based HTTP server of Brown with a push-based server that automatically pushes data, such as a file or metadata, to a client without any client request. Specifically, a POSA would not have found it obvious to replace the pull-based HTTP server taught by Brown with a push-based server that automatically pushes data to a client without any client

request. Neither the Petition provides any rationale for making such replacement. In fact, a fundamental feature of an HTTP architecture, namely, providing a service in response to a request from a client teaches away from an "automatic" provision of a service without any client request.

70. Brown discloses at 3:17-28: "The client/server architecture is designed to *minimize the load on the server processor* in order to maximize server scalability. To that end, as many processor-intensive operations as possible, such as … *synchronization itself, are performed by the client SA. Also, the polling mechanism used by the client SA* to determine if the client is synchronized with the server is designed to *require minimal processing on the server* when the client and server are in sync." Thus, because Brown aims to minimize the load on the server processor and uses a pull-based architecture, a POSA would not have been motivated to modify Brown to reach a server-push mechanism.

71. Regarding claim [3c], the Petition argues "a POSA would have found it obvious for Brown's server to automatically transmit modified files to connected clients when file updates occur," citing Hesselink, ¶¶183, 197. Pet., 68. However, just like in Brown, Hesselink uses a pull mechanism in its file synchronization process. That is, in Hesselink, the client must initiate a request for any update to the connection server.

72. A data transfer process based on this client pull mechanism is described in detail at Hesselink, ¶109: "Data from the device(s) 18 a-18 n that are destined for client computer(s) 12 a-12 n are posted to the connection server 14 a-14 n *using an HTTP request* as previously described. Receiving the HTTP request, the connection server 14 a-14 n then temporarily stores or buffers any data from the request in its memory (i.e., the sending buffer). The *next time an HTTP request comes from client computer 12 a-12 n …* the connection server 14 a-14 n retrieves the data from the sending buffer and sends the data to client computer 12 a-12 n … *along with the response for the HTTP request*."

73. This HTTP request/response process for data transfer between the client and the connection server is illustrated in FIG. 8 of Hesselink. *See e.g.*, event 890 in which "*an HTTP response* is sent to client computer 12 a-12 n by connection server 14 a-14 n," and "[i]f any data was present in the connection server sending buffer for client computer 12 a-12 n, this data is *embedded within the HTTP response*." Thus, Hesselink is focused on a data transfer process based on client pull, which is implemented by clients sending HTTP requests to the server to request the data. This is the opposite to the server initiating data push to a client.

74. The Petition acknowledges that Hesselink uses peer-to-peer but asserts that a POSA would not have been dissuaded to apply Hesselink to a client-server system because "Hesselink does not categorically disparage client/server models and

contemplates that each synchronized computer can be a "file server," citing Hesselink, ¶¶23-34, 122. However, the mere mention of a "file server" that may implement any of computers 72, 74 and 76 in FIG. 11 of Hesselink, ¶122 by no means implicates, let alone render obvious, Hesselink's applicability to a client/server model. The computers 72, 74 and 76 are part of the peer-to-peer architecture and remain as peers (*i.e.*, nodes between which a direct link is provided) regardless of whether any of these computers can be a "file server." There is nothing that suggests in Hesselink that the "file server" implements file synchronization between computers 72, 74, and 76 based on a client/server architecture. Further, while Hesselink mentions a connection server, the connection server merely serves as a pipeline or conduit in the peer-to-peer system.

75. A peer-to-peer network allows computer hardware and software to communicate between peers. Unlike a client-server architecture, there is no central server for processing requests in a peer-to-peer architecture. A peer-to-peer based file synchronization is entirely different from the client-server system because each node is the one that processes the query from another node to decide whether to grant or deny it and thus each node should have the capability to access and manage information of other nodes. A POSA would not have been motivated to combine the teachings of Brown and Hesselink because these two references teach fundamentally different architectures and are incompatible.

76. I have reviewed the secondary reference Rao and, in my opinion, it teaches the synchronization of the metadata and raw data in databases in a manner which is fundamentally different from that for which the Petition cites it. Rao ¶ 2.

77. Rao discloses the two-phase synchronization of a database, whereby, in the first phase, metadata is synchronized between a server and a client while raw data remains on the server and, in the second phase, selected raw data is synchronized to the client. Rao, ¶ 63.

78. Rao does not teach or suggest that "**the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file**" or the "**automatic[] transfer, based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device**", because Rao instead focuses on the wholesale synchronization of a database in phases – first metadata transfer and then raw data transfer. Rao does not disclose the assignment of a greater priority to the metadata than to the file, and Rao instead transfers metadata before selected raw data irrespective of any such priority. Rather, Rao categorically transfers metadata before its associated raw data.

79. Furthermore, Rao is not in the same field as that of the '787 Patent and neither it is pertinent to the problem addressed by the '787 Patent, because it focuses on synchronizing raw data in a database rather than on synchronizing files

between clients. Due to its focus on raw data in a database, a POSA would not have used Rao alone or in combination with other references with regards to the above features, "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file" or the "automatic[] transfer, based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device".

E. '823 Ground 1: The Petition Has Failed to Establish Unpatentability for Challenged Claims Based on Sigurdsson, Shappell, and Rao

80. I have reviewed the Sigurdsson reference and in my opinion, the Sigurdsson reference teaches an HTTP server architecture which is fundamentally different from the alleged suggestions that Petitioners are citing it for.

81. According to the claims of the '823 Patent, a server system is caused to: 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 (e.g., claims 1, 8 and 13)).

82. Sigurdsson does not teach or suggest that "**the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file**", because the "priority algorithm" in Sigurdsson is for prioritizing between the content that is to be synchronized and not for prioritizing between

metadata and the content. For example, in ¶58, Sigurdsson discloses three data items include 1) very large document such as a maintenance manual, 2) data from digital camera, and 3) a small office memo. Moreover, Sigurdsson discloses that, 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. As such, the server 106 can handle the higher priority data items first, and save the lower priority data items for later.

83. In FIG. 6, Sigurdsson discloses that the server receives a request for a list of missing content from a client (Client C) (612) and passes a missing content list to the client (Client C) (616) in response to an express request for a list of missing content from the client. ¶¶ 69-73. As such, the list of missing content is only provided in response to a request from the client, but there is no assignment of priority by the server. Moreover, this sequence of passing the "list of missing content" from the server 106 to Client C in step 616 in response to Client C's request for missing content in step 612 disclosed in Sigurdsson also includes step 614 in which server 106 "determines the content missing from the third client" at the time Client C requests the list of missing content. Sigurdsson, FIG. 6, ¶70. Therefore, metadata received from the first device cannot have been assigned a priority either before or upon its receipt at the server, since Sigurdsson's so-called

priority is assigned only when Client C submits a request to the server for missing content.

84. According to the claims of the '823 Patent, the server system is further caused to: **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. As such, it is the server system that "automatically transfers" the first metadata over a network to a second client device associated with the user. Sigurdsson discloses anything but a server automatically transferring the first metadata.

85. To the contrary, Sigurdsson discloses a system, in which, a receiving device (i.e., a Client C 406) requests for a list of missing content. Only in response to the server receiving the request for the list from Client C does it transmit that list of missing content to Client C. (FIG. 6, ¶¶69-73). Therefore, in Sigurdsson, the server does not automatically transfer the first metadata to the receiving device, such as Client C, since it only transfers the list of missing content when the receiving device expressly requests a list.

86. Not only does Sigurdsson fail to teach or suggest that a server **automatically transfers** the first metadata to the second client device, it also fails to teach that such a transfer is "**based on the first priority being greater than the second priority**." Sigurdsson, in FIG. 6 discloses a sequence diagram of a

request/response based mixed peer-to-peer and client/server architecture, including a server, and clients A, B, and C. In FIG. 6, the server in Sigurdsson provides a **list** of missing content only in response to a request for such a list from a client (e.g., "Client C" at steps 612 and 616).

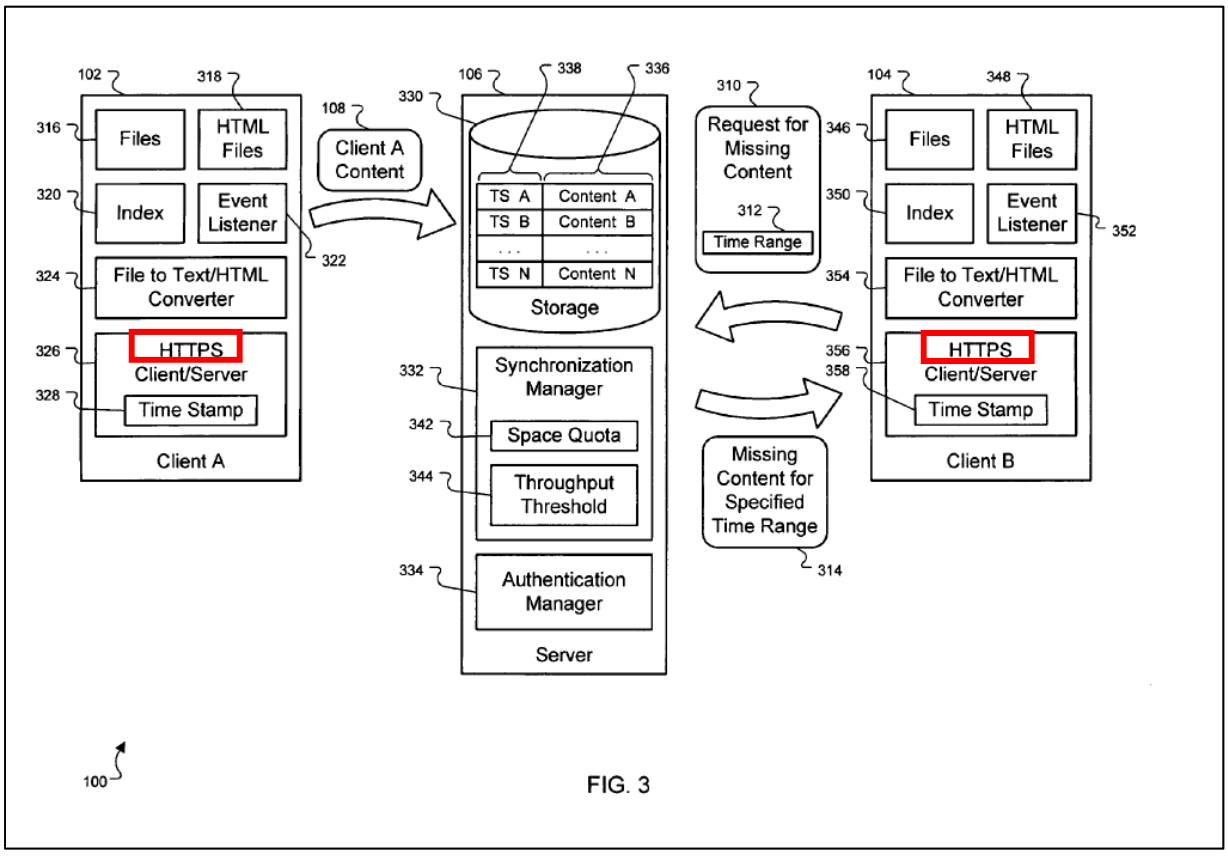
87. That is, in FIG. 6, client/server structure and operation of Sigurdsson requires an explicit request for a list of missing content from the client device ("Client C") to the server, before the server provides the list of missing content to the client device, which is neither an "**automatically transfer**" by the server nor "**based on the first priority being greater than the second priority**".

88. Sigurdsson, in ¶¶69-73 requires the following steps in order: (1) "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 …"; (2) in response, "[i]n step 614, the server determines the content missing from the third client …"; and subsequently, "[i]n step 616, the server passes the list of missing content to the third client (e.g., Client C), which had been offline." It is only after going through this sequence of Client C expressly requesting the server for list of any missing content, and the server determining the content missing from the third client, the server responds by passing the list of missing content to Client C. As such, Sigurdsson does not teach or suggest that the server 106 **automatically transfers** the list of missing content (allegedly including the first

metadata) to the Client C 406 "**based on the first priority being greater than the second priority**".

89. Since Sigurdsson discloses that the server only transfers the list of missing content *when requested* by the second client device, the transfer of the list of missing content to the second client device by the server in Sigurdsson is based on a request from the second client device and not automatically based on the first priority being greater than the second priority.

90. In FIG. 3, Sigurdsson discloses that the client devices 102 and 104 are implemented as a hypertext transfer protocol secure (HTTPS) client/server structure (*i.e.*, HTTPS Client/Server 256 and HTTPS Client/Server 356).



Sigurdsson, FIG. 3

91. HTTP (Hypertext Transfer Protocol) is a protocol used to carry requests from a browser to a Web server and to transport pages from Web servers back to the requesting browser. (Microsoft Computer Dictionary, p.259), Moreover, a HTTP server is server software that uses the HTTP protocol to serve up HTML documents and any associated files and scripts when requested by a client, such as a Web browser (Microsoft Computer Dictionary, p. 260). An HTTP/HTTPS server (e.g., the web server) plays the role of a server in a client–server model by implementing the server part of the HTTP/HTTPS network protocol. An important feature of an HTTP/HTTPS server is that the HTTP/HTTPS server waits for incoming client requests and for each request the HTTP/HTTPS server responds by providing the requested information or performing a requested task. The HTTP/HTTPS server does not perform actions on its own without a request from a client. Since the HTTP protocol is a request-response protocol, where clients send HTTP requests to the server and the server sends HTTP responses after processing each request, the HTTP protocol does not support a functionality where the server can automatically send data to a client without the client making requests for data.

92. FIG. 3 of Sigurdsson teaches a HTTPS server, which does not perform functions "automatically" without client requests for content. Thus, Sigurdsson requires a HTTPS server system based on an HTTP request/response process for

data transfer between the client and the server as illustrated in FIG. 3. Therefore, Sigurdsson not only fails to disclose the features of "**automatically transfer[ring]**, based on the first priority being greater than the second priority, **the first metadata** … to a second client device associated with the user," it would not have been obvious to modify the server system based on an HTTP request/response process of Sigurdsson to the automatically transfer the first metadata without any client request.

93. According to claim 3 (and similarly claims 10 and 15) of the '823 Patent, the computer program instructions, when executed, further **cause the server 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 … **receiving the copy of the second file from the second client device**.

94. Sigurdsson discloses a system, in which, a receiving device (i.e., a Client C 406) performs several operations before requesting a copy of missing content. Much of Sigurdsson discusses client devices requesting such missing content from other client devices, in a peer-to-peer arrangement. See Abstract. Sigurdsson describes an embodiment in which Client C obtains missing content from a server.

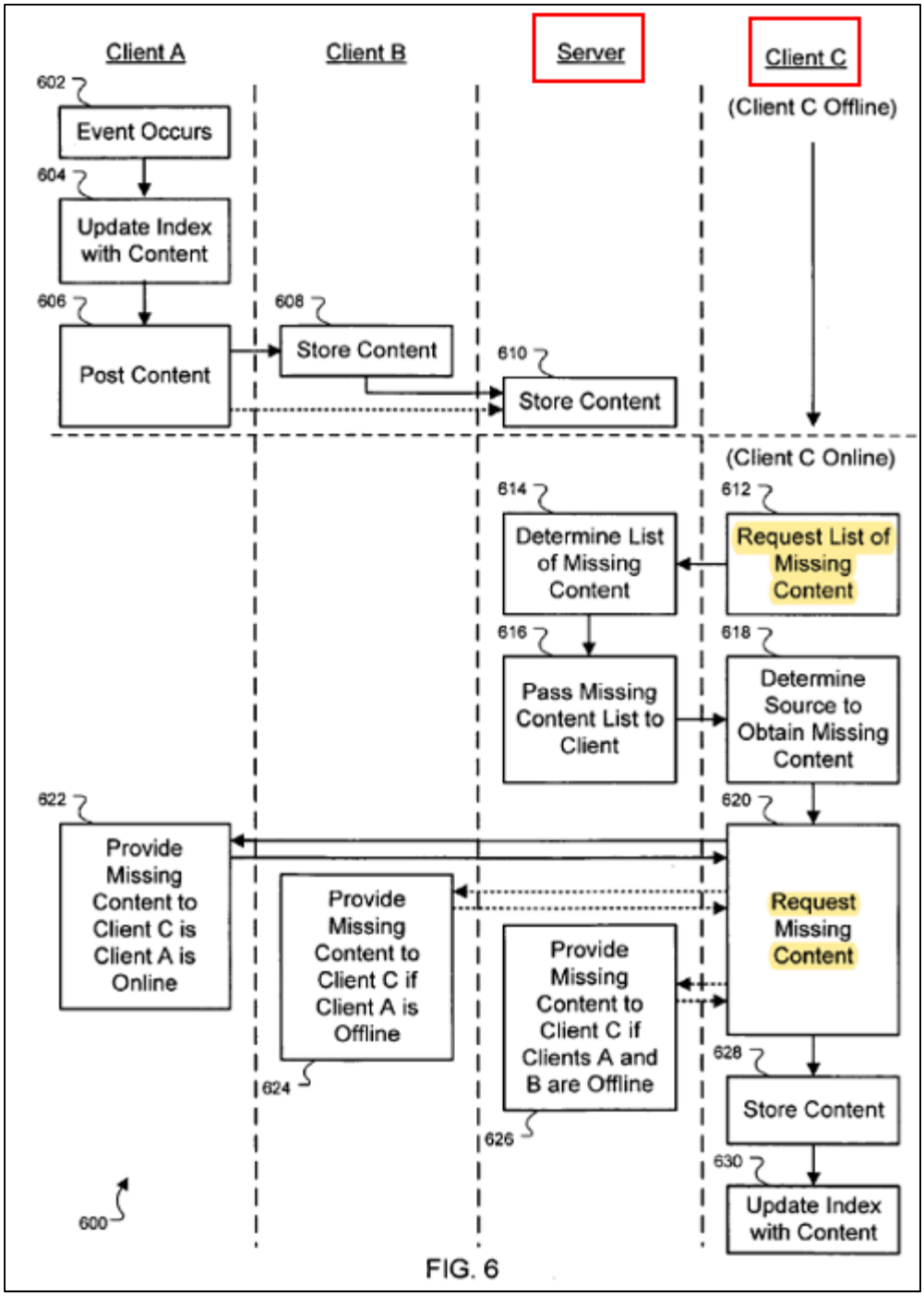
95. But that embodiment requires Client C to expressly request the missing content. Only in response to the server receiving the request from Client C does it

transmit that content to Client C. (FIG. 4, 6, ¶¶71-75, 80). Therefore, in Sigurdsson, the server does not automatically transfer a modified copy of the file to the receiving device, such as Client C, since it only transfers the missing content when the receiving device expressly requests it.

96. The automatic transfer in the claims of the '823 Patent refers to the server system automatically transferring the copy of the first file.

97. Not only does Sigurdsson fail to teach or suggest that a server "**automatically transfer[s]** the copy of the second file to the first client device associated with the user ", it also fails to teach that such a transfer is "**responsive to** … **receiving the copy of the second file from the second client device**." Sigurdsson's transfer to Client C is the result of Client C requesting the transfer.

98. Sigurdsson, in FIG. 6 (reproduced below with annotations), discloses a sequence diagram of a request/response based mixed peer-to-peer and client/server architecture, including a server, and clients A, B, and C. As shown in FIG. 6, the server in Sigurdsson provides missing content only in response to a request from a client (e.g., "Client C" at steps 612 and 620). That is, in FIG. 6, client/server structure and operation of Sigurdsson requires at least two separate requests from the client device ("Client C") to the server, before the server provides the missing content to the client device, neither of which are "receiving the copy of the first file from the first client device".



Sigurdsson, FIG. 6

99. It is only after going through this sequence of Client C expressly requesting the server for any missing content, Client C determining where it can obtain that

missing content, and then requesting the content from the server, before the server responds by transferring the content to Client C. As such, Sigurdsson does not teach or suggest that the server 106 automatically transfers the content 410 to the Client C 406 responsive to receiving the content 410 from Client A 402.

100. Sigurdsson at FIG. 2 and ¶¶ 37, 40 and 46 discloses a request for missing content from the second client. For instance, "**the second client polls** for updates" (¶37), the request is "**periodically issue[d]", and "the request 310 includes a Time Range 312 parameter** identifying the range of time stamps for the missing content" ¶40 and "the Request 310 is issued **in order** for the Client B 104 to obtain the Client A Content 108 that the Server 106 has yet to send to the Client B104" (¶46).

101. FIG. 5 and the related text of Sigurdsson does not disclose several of the claimed features including the "**automatically transfer** the copy of the second file to the first client device", "responsive to … **(ii) receiving the copy of the second file from the second client device**" as discussed above." In fact, the disclosure in FIG. 5 simply summarizes the request-based file transfer approach from FIG. 6 discussed above in detail.

102. As discussed above, FIG. 3 of Sigurdsson teaches a HTTPS server, which does not perform functions "automatically" without client requests for content. Thus, Sigurdsson requires a HTTPS server system based on an HTTP

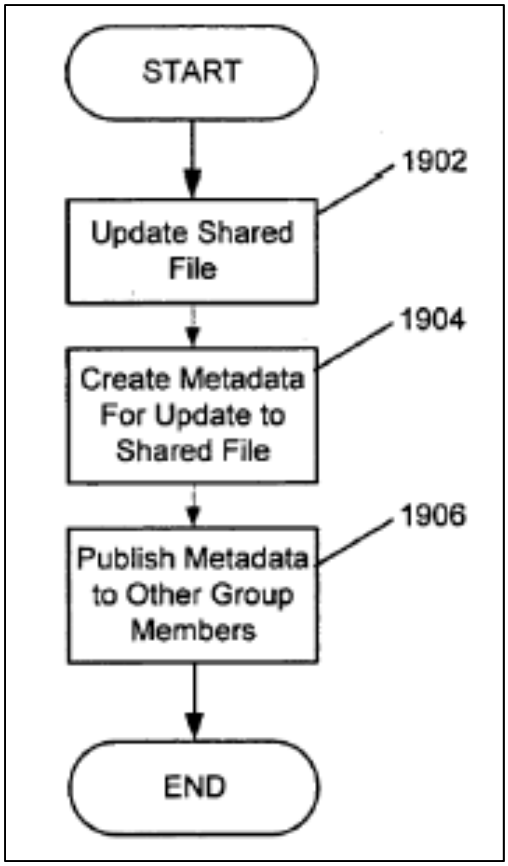
request/response process for data transfer between the client and the server as illustrated in FIG. 3. Therefore, Sigurdsson not only fails to satisfy the features of "**automatically transfer** the copy of the second file to the first client device", it would not have been obvious to replace the principal structure (i.e., the server system based on an HTTP request/response process) taught by Sigurdsson with the automatically transfer of a copy of the file to a client without any client request to disclose the claim features.

103. I have reviewed the secondary reference Shappell and in my opinion it discloses a peer-to-peer network environment which is fundamentally different from a Client-Server environment. ¶ 7 of Shappell. The Abstract of Shappell indicates that the disclosed system and methods disclosed are used to enable users to share files in a server-less shared space. By emphasizing a "server-less" environment, Shappell does not teach a Client-Server system that uses a server.

104. Shappell discloses "a computer implemented method and system enable users to share files in a **server-less** shared space. By providing access to such **server-less** spaces via a visual presentation, the system renders content available for access by other group members. Access is sometimes provided through propagation of metadata or other uniquely identifying indicia associated with the shared space to all group members." Abstract. Shappell teaches away from using a server, Shappell does not teach or suggest a server system **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 receiving the copy of the first file from the first client device**.

105. Shappell is silent about a priority being assigned to the first metadata or the copy of the first file. In particular, in cited FIG. 19 (reproduced below), Shappell at most discloses "creating metadata for update to shared file" and "publishing metadata to other group members." However, there is no mention of a priority being assigned to the metadata. No other teaching in Shappell discloses a priority being assigned to the metadata. Therefore, Shappell does not teach or suggest "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file".



Shappell (EX 1020), FIG. 19

106. In FIG. 19, step 1906, Shappell discloses the metadata to all other group members, which is different from the claimed features of automatically transferring the first metadata **to the second client device associated with the same user**.

107. I have reviewed the secondary reference Rao and, in my opinion, it discloses the synchronization of the metadata and raw data in databases in a manner which is fundamentally different from that for which the Petition cite it. Paragraph 0002 of Rao.

108. Rao discloses the two-phase synchronization of a database, whereby, in the first phase, metadata is synchronized between a server and a client while raw data

remains on the server and, in the second phase, selected raw data is synchronized to the client. Paragraph 0063 of Rao.

109. Rao does not teach or suggest that "**the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file**" or the "**automatic[] transfer, based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device**", because Rao instead focuses on the synchronization of a database in phases – first metadata transfer and then raw data transfer. Rao does not disclose the assignment of a greater priority to the metadata than to the file, and Rao instead transfers metadata before selected raw data irrespective of any such priority. Rather, Rao categorically transfers metadata before its associated raw data.

110. Furthermore, Rao is not in the same field as that of the '823 Patent and neither it is pertinent to the problem addressed by the '823 Patent, because it focuses on synchronizing raw data in a database rather than on synchronizing files between clients. Due to its focus on raw data in a database, a POSA would not have used Rao alone or in combination with other references with regards to the above features, "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file" or the "automatic[] transfer, based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device".

F. '823 Ground 2: The Petition Has Failed to Establish Unpatentability for *Challenged Claims* Based on Brown, Hesselink, Shappell, and Rao

111. The Petition argues that Brown describes that the originating "first client device" sends to the server "CSI" (Brown, 10:9-10) and "parent directory SID," and "file name," *id*., 15:38-40, 5:50-52), which allegedly corresponds to metadata for a modified file, prior to the modified file being uploaded, citing *id.*, 13:57-62 and Fig. 8A (step 805), 14:5-6 and Fig. 8B (step 850), Figs. 11A-C and 15:30-54 (step 1115 and step 1120). Pet., 55-56.

112. However, the client sync index ("CSI") in Brown merely refers to a value indicating whether the client directory is up to date ("If the SSI value matches the CSI value passed in by the polling client, the client is up to date with the current state of its server account."; Brown, 5:17-19). The CSI received from a client is not "associated with the updated version" of the file that is generated from the user modifying the content of the file, as recited in claims [1c], [8b], and [13b]. Also, in Brown, the CSI is included in the sync poll call from the client Synchronization Application (SA) to the server, and the server returns its server sync index (SSI) back to the client in response. That is, the CSI as disclosed in Brown is merely a sync call requesting the server's SSI so that the client can decide whether the CSI matches with the SSI (that is, whether the client directory is up to date).

113. The Petition also relies upon step 1115 of FIG. 11A of Brown, which discloses that the client sends the SID, the parent directory SID, and the file name to the server, and argues that this portion allegedly teaches "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file" in claim [1c]. However, the SID, the parent directory SID, and the file name are transmitted to the server only for the purpose of locating, if any, a previous version of the file is on the server, at step 1117 in FIG. 11B. If there is no previous version of the file on the server, then at step 1120, *the file and the client metadata item* are uploaded to the server.

114. The Petition, apparently with regards to claim [1d], cites to the server synchronization data ("SSD") in Brown as the metadata that the server first transfers to the receiving client before the server transfers a copy of the modified file to that receiving client. Pet., 56. However, the SSD contains server metadata items for any items with file sync index (FSIs) greater than the CSI passed in by the receiving client's sync poll call. Brown, 8:53-61. That is, the SSD can be generated *only after the CSI is received* from the receiving client's sync poll call. This is also consistent with Brown's disclosure that the SSD is generated responsive to a client sync index (CSI) received from the client. Brown, claims 9-10.

115. Further, because the SSD needs to include server metadata items for any items that are updated from the last sync call (that is, any items with FSIs greater than the client's current CS), Brown necessarily requires the client metadata received at step 1120 in FIG. 111B in order to generate the SSD.

116. Therefore, the "SSD" of Brown does not teach "the first metadata that is *automatically* transferred to a second client device *based on the first priority being greater than the second priority*."

117. In Brown, the client SA needs to poll the server to check if there are any changes in the client's directory, and if the client's directory is not up to date, the client then requests server synchronization data (SSD) to learn about the new changes that have been made at server 105. ("If the CSI matches the server account's SSI value, then the client is up to date with the server. Otherwise *the client SA requests server synchronization data (SSD)*."; Brown, 8:51-53.

118. If 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." Brown, 10:19-21. Based on the SSD received from the server, "the *client SA updates the client's directory and metadata* to match the server state." Brown, 8:62-63. That is, in Brown, the SSD are transmitted to the client device *upon the request received from*

the client. *See e.g.,* Brown, 10:18-27 (the server sends the SSD to client 130 *in response to a request for the SSD by the client*).

119. Brown teaches a HTTP server, which uses a client-pull architecture and does not perform functions "automatically" without client requests. Brown, 4:12-15 ("[C]ommunication between the clients and server 105 is tunneled within the hypertext transport protocol (HTTP).").

120. As discussed above, the HTTP protocol is a request-response protocol, and the HTTP servers use the client-pull architecture, where clients send HTTP requests to the server and the server sends HTTP responses after processing each request. The HTTP protocol does not support push-based functionality where the server can automatically send data to a client without the client making requests for data.

121. Because Brown requires a HTTP server system based on an HTTP request/response process for synchronization, Brown fails to disclose the features related to automatically transferring the first metadata, as recited in claims [1d], [8c], and [13c], or automatically transferring the copy of the second file that is modified.

122. It would not have been obvious to replace the pull-based HTTP server of Brown with a push-based server that automatically pushes data, such as a file or metadata, to a client without any client request. Specifically, a POSA would not

have found it obvious to replace the pull-based HTTP server taught by Brown with a push-based server that automatically pushes data to a client without any client request. Neither the Petition provides any rationale for making such replacement. In fact, a fundamental feature of an HTTP architecture, namely, providing a service in response to a request from a client teaches away from an "automatic" provision of a service without any client request.

123. Brown discloses at 3:17-28: "The client/server architecture is designed to *minimize the load on the server processor* in order to maximize server scalability. To that end, as many processor-intensive operations as possible, such as … *synchronization itself, are performed by the client SA*. Also, the *polling mechanism used by the client SA* to determine if the client is synchronized with the server is designed to *require minimal processing on the server* when the client and server are in sync." Thus, because Brown aims to minimize the load on the server processor and uses a pull-based architecture, a POSA would not have been motivated to modify Brown to reach a server-push mechanism.

124. Regarding feature [3c], the Petition argues that a POSA would have been motivated to combine Brown with Hesselink "to automatically transmit modified files to connected clients as soon as file updates occur." Pet., 69. However, just like in Brown, Hesselink uses a pull mechanism in its file synchronization process.

That is, in Hesselink, the client must initiate a request for any update to the connection server.

125. A data transfer process based on this client pull mechanism is described in detail at Hesselink, ¶109: "Data from the device(s) 18 a-18 n that are destined for client computer(s) 12 a-12 n are posted to the connection server 14 a-14 n *using an HTTP request* as previously described. Receiving the HTTP request, the connection server 14 a-14 n then temporarily stores or buffers any data from the request in its memory (i.e., the sending buffer). The *next time an HTTP request comes from client computer 12 a-12 n* … the connection server 14 a-14 n retrieves the data from the sending buffer and sends the data to client computer 12 a-12 n … *along with the response for the HTTP request*."

126. This HTTP request/response process for data transfer between the client and the connection server is illustrated in FIG. 8 of Hesselink. *See e.g.*, event 890 in which "*an HTTP response* is sent to client computer 12 a-12 n by connection server 14 a-14 n," and "[i]f any data was present in the connection server sending buffer for client computer 12 a-12 n, this data is *embedded within the HTTP response*." Thus, Hesselink is precisely focused on a data transfer process based on client pull, which is implemented by clients sending HTTP requests to the server to request the data. This is the opposite to the server initiating data push to a client.

127. The Petition acknowledges that Hesselink uses peer-to-peer but asserts that a POSA would not have been dissuaded to apply Hesselink to a client-server system because "Hesselink does not categorically disparage client/server models and contemplates that each synchronized computer can be a "file server," citing Hesselink, ¶¶23-34, 122. However, the mere mention of a "file server" that may implement any of computers 72, 74 and 76 in FIG. 11 of Hesselink, ¶122 by no means implicates, let alone render obvious, Hesselink's applicability to a client/server model. The computers 72, 74 and 76 are part of the peer-to-peer architecture and remain as peers (*i.e.*, nodes between which a direct link is provided) regardless of whether any of these computers can be a "file server." There is nothing that suggests in Hesselink that the "file server" implements file synchronization between computers 72, 74, and 76 based on a client/server architecture.

128. Further, while Hesselink mentions a connection server, the connection server merely serves as a pipeline or conduit in the peer-to-peer system.

129. A peer-to-peer network allows computer hardware and software to communicate between peers. Unlike a client-server architecture, there is no central server for processing requests in a peer-to-peer architecture. A peer-to-peer based file synchronization is entirely different from the client-server system because each node is the one that processes the query from another node to decide whether to

grant or deny it and thus each node should have the capability to access and manage information of other nodes. A POSA would not have been motivated to combine the teachings of Brown and Hesselink because these two references teach fundamentally different architectures and are incompatible.

130. I have reviewed the secondary reference Rao and, in my opinion, it discloses the synchronization of the metadata and raw data in databases in a manner which is fundamentally different from that for which the Petition cites it. Paragraph 0002 of Rao.

131. Rao discloses the two-phase synchronization of a database, whereby, in the first phase, metadata is synchronized between a server and a client while raw data remains on the server and, in the second phase, selected raw data is synchronized to the client. Paragraph 0063 of Rao.

132. Rao does not teach or suggest that "**the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file**" or the "**automatic[] transfer, based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device**", because Rao instead focuses on the synchronization of a database in phases – first metadata transfer and then raw data transfer. Rao does not disclose the assignment of a greater priority to the metadata than to the file, and Rao instead

transfers metadata before selected raw data irrespective of any such priority. Rather, Rao categorically transfers metadata before its associated raw data.

133. Furthermore, Rao is not in the same field as that of the '823 Patent and neither it is pertinent to the problem addressed by the '823 Patent, because it focuses on synchronizing raw data in a database rather than on synchronizing files between clients. Due to its focus on raw data in a database, a POSA would not have used Rao alone or in combination with other references with regards to the above features, "the first metadata being assigned a first priority greater than a second priority assigned to the copy of the first file" or the "automatic[] transfer, based on the first priority being greater than the second priority, [of] the first metadata over a network to a second client device".

IV. CONCLUSION

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 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 section 1001 of Title 18 of the Unites States Code.

Respectfully submitted,

Prashant Shenoy, PhD.

Date: May 15, 2023