IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

_____

BEFORE THE PATENT TRIAL AND APPEAL BOARD

_____

NETFLIX, INC.,

Petitioner,

v.

AVAGO TECHNOLOGIES INTERNATIONAL SALES PTE. LIMITED,

Patent Owner.

_____

Case No. IPR2020-01582

U.S. Patent 8,572,138

_____

Declaration of Prashant Shenoy

Table of Contents

I, Prashant Shenoy, declare as follows:

1. My name is Prashant Shenoy. I am a Distinguished Professor of Computer Science and Director of the Center for Smart and Connected Society at the University of Massachusetts. I have prepared this report as an expert witness retained by Netflix, Inc. In this report I give my opinions as to whether certain claims of U.S. Patent No. 8,572,138 ("the '138 patent") are invalid. I provide technical bases for these opinions as appropriate.

2. This report contains statements of my opinions formed to date and the bases and reasons for those opinions. I may offer additional opinions based on further review of materials in this case, including opinions and/or testimony of other expert witnesses. I make this declaration based upon my own personal knowledge and, if called upon to testify, would testify competently to the matters contained herein.

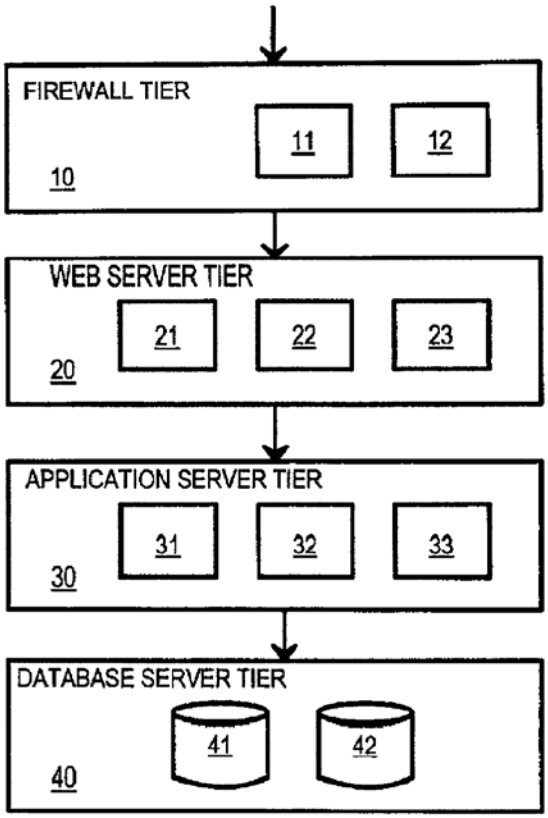
I. OVERVIEW OF THE TECHNOLOGY

3. For decades, websites such as e-commerce sites, online retail stores, and financial services sites, have been implemented with distributing computing systems. In a distributed computing system, multiple computers are connected over a network, and applications are assigned to computers within the network, which are also referred to as servers, nodes, or hosts.

4. By the early 2000s, it was common for nodes to be organized into tiers, and automation systems have long been used to deploy new instances of applications to nodes, in order to scale the capacity of the system to match user demand for a website. While virtualization was invented in the 1960s, it became common in the early 2000s for distributed computing systems to use virtual machines for existing tiered architectures and to automate their deployment using rules engines.

A. Multi-Tiered Infrastructure for Websites

5. Internet applications and web services often employed a multi-tier architecture, with each tier containing numerous servers that were often referred to as "nodes." *See* Ex. 1009 at 1; Ex. 1007 ¶[0039]. A common design for the multi-tier architecture employed a three-tier model, with nodes organized into a web server tier, an application server tier, and a database tier. *See* Ex. 1007 ¶¶[0024]-[0039], FIG. 2; Ex. 1009 at 3; Ex. 1004 23:11-20. Some applications included an additional firewall tier for security. *See, e.g.*, Ex. 1007, Fig. 1:



6. A multi-tier model allowed for improved monitoring, diagnosis, and automated control of the various software components of a website, which helped meet the service-level objectives ("SLOs") specified in service-level agreements ("SLAs"). *See* Ex. 1007 ¶[0027]; Ex. 1009 at 3-4. Therefore, it was common to monitor the status of nodes, including by service and tier, to diagnose and correct problems such as availability and performance bottlenecks, for example when the demand for a website exceeded its current capacity. *See* Ex. 1007 ¶¶[0037]-[0038]. Automated processes were often used to respond to issues to ensure that SLOs were met, as explained below.

B. Automatic Deployment of Applications

7. Large business applications, such as enterprise or Web applications, needed to scale up and down to match demand and meet SLOs. *See, e.g.*, Ex. 1009 at 4; Ex. 1007 ¶[0027]. Applications could be scaled up by deploying new software instances in the appropriate tier. *See* Ex. 1009 at 4; Ex. 1007 ¶[0072], Abstract. For example, new instances could be deployed in the appropriate tiers to double capacity during a flash crowd. *See* Ex. 1009 at 1, 12.

8. Automated systems were often used to automatically deploy new application instances to nodes. *See, e.g.*, *See* Ex. 1009 at 1, 4-5. By the 1990s and early 2000s, it was common for automated systems to use knowledge-based systems, such as rules engines, to select an appropriate node based on the monitored status of nodes. *See* Ex. 1013 ¶¶[0083]-[0084], [0163]-[0164], FIGS, 5, 10; Ex. 1006, 3:44-58; Ex. 1001, 26:40-61; Ex. 1010 at 286. Such systems and rule engines, including forward-chaining rule engines employing algorithms such as LEAP or RETE, were known in the art. *See id.*

9. There were two types of rules engines: forward-chaining and backward-chaining rules engines. Forward-chaining algorithms applied Modus Ponens inference rules and were commonly used for systems that acted in response to new information. *See* Ex. 1010 at 280, 310. For automated deployment of distributed applications, forward-chaining rules were common because they could

respond to new status information collected from nodes, including for example by deploying new instances of constrained service components. *See, e.g.*, Ex. 1007 ¶¶[0071]-[0072]. In contrast, backward-chaining algorithms were more common for optimization scenarios because they started with a goal and searched backwards to determine rules or actions that satisfied the goal.

10. To facilitate automated deployment, application images were often stored in a repository. Using a repository allowed efficient management, typically using a database, of the different application images that were available, including different versions and configurations. *See, e.g.*, Ex. 1004, 12:13-23; Ex. 1005, 60:64-61:8. When new instances of particular software applications were needed, the repository provided an efficient and organized system for locating and deploying the application.

C. Virtual Machines

11. A virtual machine ("VM") is a software abstraction (virtualization) of a physical machine. IBM invented virtualization in the 1960s and has applied the technology for decades. In the early 2000s, VMs became popular for distributed computing platforms, including for websites, because they allowed a single physical server to act as multiple virtual servers by hosting multiple VMs. Software could run inside a VM without ever knowing that it was running inside a virtualized computer platform. Therefore, VMs were easily relocatable and

replicable to other machines and systems, which made them well-suited for the automated deployment and scaling of distributed computing systems. *See* Ex. 1005, 11:42-54; Ex. 1004, 8:13-20.

12. A key component for virtualization was the virtual machine monitor ("VMM"), also referred to as a "hypervisor," which was a piece of software that ran on a host machine and virtualized the resources of the machine for VMs. *See, e.g.*, Ex. 1004, 7:7-27, Ex. 1005 12:33-44. There are two types of VMMs: a Type 1 VMM, also referred to as a hypervisor OS, ran directly on the hardware and acted as the operating system, while a Type 2 VMM ran as an application on top of a separate host OS, such as Windows or Linux. *See, e.g.*, Ex. 1004, 7:7-27, 1005, 66:2-10. VMware was, and still is, an industry leader in the virtualization. Its ESX software featured a Type 1 VMM that ran directly on the hardware, while its GSX software featured a Type 2 VMM that ran on top of a separate OS. *See* 1005, 66:2-10.

13. Because the VM virtualizes a physical computer, the software image of the VM contains the state of the virtual computer. The VMM can suspend VMs by stopping the execution of VM programs and saving the VM state to its VM image. VM images can be replicated, transferred, and resumed.

14. Virtualization simplified the management of distributed computer systems because the combination of an application running inside a VM could be

saved together as a VM image. VM images were stored in repositories. *See, e.g.*, Ex. 1004, 10:27-29, 17:30-50, 18:1-15. To deploy an application, the VM image for that application can be copied from a repository to a different node and then the VM can be resumed. By 2005, virtual machines were widely used in distributed computing applications, with commercial software available from VMware, IBM, and others.

II. THE ’138 PATENT

A. Overview

15. According to the ’138 patent, “[o]ne challenge with distributed computing systems is the organization, deployment and administration of such a system within an enterprise environment.” Ex. 1001, 1:26-33. “For example, it is often difficult to manage the allocation and deployment of enterprise computing functions” which “often includes several business groups, and each group may have competing and variable computing requirements.” *Id.*

16. The ’138 patent purports to solve these difficulties with “a distributed computing system that conforms to a multi-level, hierarchical organizational model.” Ex. 1001, 1:37-43. In embodiments of the ’138 patent, the tiers include “a web server tier … a business application tier … and a database tier …” Ex. 1001, 13:27-56, FIG. 2. The alleged invention also “includes a software image repository” that stores image instances of virtual machine managers and software

applications, and a "control node that comprises an automation infrastructure to provide autonomic deployment of the image instances…" *Id.*, 2:15-30. The automation infrastructure may use a "forward chaining" rule engine "that performs logical reasoning using knowledge encoded in high-level condition-action rules" to provide automated reasoning. *See* Ex. 1001, 22:20-62.

17. The '138 patent recites virtual machines but does not purport to invent them. Instead, the embodiments of the '138 patent rely on commercially available software and image formats from VMware. *See, e.g.*, Ex. 1001, 32:59-64 ("For example, virtual machine manager 402 may be an instance of VMware ESX software produced by VMware, Inc. of Palo Alto, Calif."), 33:25-41 ("For instance, if virtual machine manager 402 is an instance of VMware ESX, virtual machine disk image files 408 may be specially captured .vmdk files.").

18. Indeed, the alleged invention of the '138 patent was already known in the art and taught, for example, by VMware prior art that describes a central repository for virtual machine images and software applications, where a "heuristic/rules engine" automatically deploys images over the network to distributed host computers. *See, e.g., infra* §III.B; *see also* Ex. 1004, Abstract, 10:27-42; Ex. 1005, 59:21-54.

19. While the challenged claims recite various details regarding the organization of nodes into tiers and a forward-chaining rules engine, those

limitations were directed to well-known features of prior art distributed systems. The tiers disclosed by the '138 patent were widely known and typical for web sites. *See, e.g.*, Ex. 1007, Fig. 2, ¶¶[0024]-[0039]; Ex. 1009 at 1, 3; *supra* §I.A; *infra* §VI.C. The '138 patent does not purport to have invented forward-chaining rules engines and instead incorporates by reference several prior art papers on the topic. Ex. 1001, 26:26-61; *see also, e.g.*, Ex. 1013 ¶¶[0083]-[0084], [0163]-[0164]; Ex. 1006, 3:44-58; Ex. 1004, 10:27-24. The alleged invention is nothing more than a combination of prior art elements according to known methods to yield predictable results.

B. Priority Date

20. The application for the '138 patent was filed on March 30, 2007, claiming priority to provisional application No. 60/787,280, filed on March 30, 2006.[1] Columns 8:24 to 13:26 of the '138 patent, which contain disclosures relating to an infrastructure management facility ("IMF"), were not part of the provisional application. *See* Ex. 1001, 8:24-13:26.

[1] I do not analyze whether the '138 patent is entitled to claim priority to the provisional but use 2006 for simplicity because all the relied-upon prior art predates 2006.

C. Challenged Claims

21. I understand that Petitioner is challenging the validity of claims 1-5, 9-10, 17, 19-21, and 26 of the ’138 patent in the Petition for Inter Partes Review to which this declaration will be attached. Those claims are reproduced in Appendix 3. While my declaration is directed to the challenged claims, I have considered all claims of the ’138 patent, as well as portions of the ’138 patent prosecution history in forming my opinions.

III. SUMMARY OF THE PRIOR ART

22. There are a number of patents and publications that constitute prior art to the ’138 patent. I have reviewed and considered the prior art discussed in this section, along with the materials listed in Appendix 2.

A. Invalidity Grounds

23. Based on my review and analysis of the materials cited herein, my opinions regarding the understanding of a person of ordinary skill in the art (“POSITA”) in the 2006 timeframe, and my training and experience, it is my opinion that the challenged claims of the ’138 patent are invalid based on the following grounds:

Grounds	Claims	Basis	Prior Art
1	1-5, 9, 19-21, and 26	Obviousness	Khandekar
2	1-5, 9, 19-21, and 26	Obviousness	Khandekar in view of Le
3	9, 10	Obviousness	Khandekar in view of Le and Sidebottom

4	17	Obviousness	Khandekar in view of Le and Stone

24. I understand that none of the above-referenced art was cited or considered during prosecution of the ’138 patent. First, none of these references or any related patents are listed on the face of the ’138 patent. In addition, I reviewed the file history for the ’138 patent and am not aware of any of the above references being discussed in any office action or in the prosecution history generally.

B. U.S. Patent No. 7,577,722 (“Khandekar”) (Exhibit 1004)

25. Khandekar was filed on April 5, 2002 and issued August 18, 2009. Khandekar is prior art because it was filed before the earliest possible priority date of March 30, 2006 for the ’138 patent.

26. Khandekar is a VMware patent that teaches a repository, called a provisioning server, for storing pre-configured, ready-to-deploy VM images, with operating systems and applications installed, and then automatically deploying those VM images to host computers in a distributed computing system. *See, e.g.*, Ex. 1004, 9:26-27 (“The main feature of the invention is a VM provisioning server…”), 4:30-35, 5:25-36, 10:15-26, 23:54-24:27, Abstract. To facilitate automated deployment, Khandekar teaches a “heuristics/rules engine” that automatically selects and deploys VM images to host computers based on the monitored status of the hosts. *See* Ex. 4:58-65, 10:27-43. Khandekar teaches that a

VMM may be included with the VMs that are deployed to hosts. *See, e.g.*, Ex. 1004 8:36-44, 13:17-35 ("a VMM is typically included for each VM in order to act as its software interface with the underlying host system"). Khandekar teaches each VMM supporting a single VM, or multiple VMs. *See, e.g.*, 1004, 7:28-38.

C. U.S. Patent No. 8,209,680 ("Le") (Exhibit 1005)

27. Le was filed on June 30, 2003, issued June 12, 2012, and claims priority to provisional application 60/462,445 filed on April 11, 2003. Le is prior art because it is filed before the earliest possible priority date for the '138 patent.

28. Le is a VMware patent that teaches techniques for managing distributed networks of physical computers and virtual machines. *See, e.g.*, Ex. 1005, 59:29-36. Le teaches the use of VMware software—the same software relied upon by the '138 patent's embodiments—and various techniques for capturing, configuring and deploying images amongst physical and virtual computers. *See, e.g.*, Ex. 1005, Abstract. Le also teaches agents that monitor the status and health of deployed machines. *See* Ex. 1005, 68:21-24.

D. U.S. Patent No. 7,962,633 ("Sidebottom") (Exhibit 1006)

29. Sidebottom was filed on October 13, 2005 and issued June 14, 2011. Sidebottom is prior art because it is filed before the earliest possible priority date for the '138 patent.

30. Sidebottom teaches a rules engine that applies a forward-chaining rules algorithm to monitored events on a network, using condition/action rules to trigger automated responses. *See, e.g.*, Ex. 1006, 3:32-58, 5:63-6:17, 4:26-44, 9:36-45.

E. U.S. Patent Application Publication No. 2003/0036886 ("Stone") (Exhibit 1007)

31. Stone was filed on August 20, 2001 and published February 20, 2003. Stone is prior art because it is published more than one year before the earliest possible priority date for the '138 patent.

32. Stone "relates to distributed computer systems, and more particularly to performance monitoring and control of server computers and applications." Ex. 1007 ¶[0001]. Stone teaches organizing nodes into "web server," "application," and "database" tiers, with "node monitors" that monitor events by tier. *See, e.g.*, Ex. 1007 ¶[0040], FIG. 2. The monitoring allows an SLO agent to take actions if an SLO is not met, such as replicating more of the constraining service components or nodes or increase resources. *See* Ex. 1007 ¶[0057], Abstract.

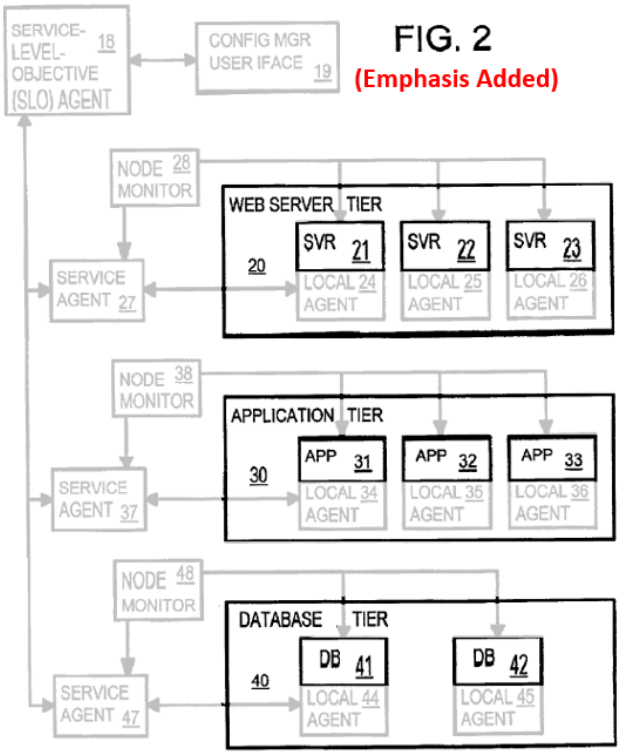


FIG. 2
(Emphasis Added)

IV. LEVEL OF ORDINARY SKILL IN THE ART

33. In determining the characteristics of a hypothetical POSITA of the '138 patent at the time of the claimed invention, I considered several things, including various prior art techniques relating to the field of computing environments and distributed computing systems, the type of problems that such techniques gave rise to, and the rapidity with which innovations were made. I also considered the sophistication of the technologies involved, and the educational background and experience of those actively working in the field at the time. I also considered the level of education that would be necessary to understand the '138 patent. Finally, I placed myself back in the relevant period of time and considered

the engineers, programmers, and researchers that I have worked with in the field of computing environments and distributed computing systems.

34. I came to the conclusion that a person of ordinary skill in the field of art of the ’138 patent would have been a person with a bachelor’s degree in electrical engineering, computer science, or a similar field with at least two years of experience in distributed computing systems or a person with a master’s degree in electrical engineering, computer science, or a similar field with a specialization in distributed computing systems. A person with less education but more relevant practical experience may also meet this standard.

35. In view of the types of problems encountered in the art and prior art solutions to those problems (*see supra* §I), including the widespread use of virtualized systems and popularity of commercial virtual machine software, a POSITA in 2006 with the level of skill described above would have had the knowledge and skills necessary to:

- Create a distributed computing system with a network of computing devices/servers;
- Use virtualized computing systems, including the use of commercially available software from VMware such as ESX and GSX software;
- Create images of physical and virtual machines, including images that include VMMs, operating systems, and applications;

- Use rules engines and other knowledge-based systems to automatically deploy images to the nodes of a distributed computing system;
- Implement multi-tier applications and web services on distributed computer systems; and
- Implement systems to monitor the status and state of distributed computer systems.

V. CLAIM CONSTRUCTION

36. For purposes of this inter partes review, I have considered the claim language, specification, and portions of the prosecution history, to determine the meaning of the claim language as it would have been understood by a person of ordinary skill in the art at the time of the invention. It is my understanding that the "plain and ordinary meaning" has traditionally been applied in district court litigation, where a claim term is given its plain and ordinary meaning in view of the specification from the view point of a person of ordinary skill in the art. I understand that this is referred to as the *Philips* standard.

37. I have applied the Philips standard in my analysis. Unless otherwise stated, I have applied the plain and ordinary meaning to claim terms.

A. "a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines" (Claim 1)

38. A POSITA would have understood that the term "a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines" is satisfied by, but is not necessarily limited to, a plurality of virtual disk image files. Dependent claim 2 specifies that "the image instances of the plurality of software applications comprise virtual disk image files." Since claim 2 depends on claim 1, a teaching that is sufficient to satisfy claim 2 must also be sufficient to satisfy claim 1.

39. The '138 patent includes embodiments where "[t]he operating system data and application data, as well as instance-specific configuration data for the underlying virtual machines 404, is stored as respective virtual disk image files 408." Ex. 1001, 33:25-30, 33:55-67, 5:12-22, 5:40-56. The specification explains that a computer may be configured with an operating system and application data before the image is captured. *See id.*

40. It is my understanding that it is improper to import limitations from the embodiments into the claims. Nonetheless, to the extent Patent Owner argues that this claim term requires the virtual disk image files to include an operating system and application data, it is taught by the prior art as explained below.

VI. INVALIDITY

41. Based on my review and analysis of the materials cited herein, my opinions regarding the understanding of a POSITA in the 2006 timeframe, and my training and experience, it is my opinion that a POSITA would have found claims 1-5, 9-10, 17, 19-21, and 26 of the ’138 patent obvious. I understand that the Petitioner is not aware of any claim by the Patent Owner that the ’138 patent is entitled to an earlier priority date. My opinions regarding obviousness would not materially change if the challenged claims were entitled to a priority date within roughly half a year earlier. The reasons for my conclusions are explained more fully below.

A. Grounds 1 and 2: Claims 1-5, 9, 19-21, and 26 are Rendered Obvious by Khandekar Alone (Ground 1), and Khandekar in View of Le (Ground 2)

42. It is my opinion that a POSITA would have found Claims 1-5, 9, 19-21, and 26 obvious based on the teachings of Khandekar by itself. Le, another reference by the same inventors and assignee VMware, reinforces Khandekar’s teachings with implementation details regarding, for example, VMware software, the ratio of VMMs to VMs, and disk imaging techniques.

43. For example, Khandekar teaches that each VMM may support a single VM, or each VMM may support multiple VMs. *See, e.g.*, 1004, 7:28-38. Khandekar therefore teaches a VMM providing a plurality of VMs. To the extent it

is argued otherwise, Le teaches that VMM providing a plurality of VMs. Le further teaches implementation details regarding VMware software and techniques for creating and deploying images to physical and virtual computers. Le's teachings complement Khandekar's teachings, and the combination further renders the challenged claims obvious.

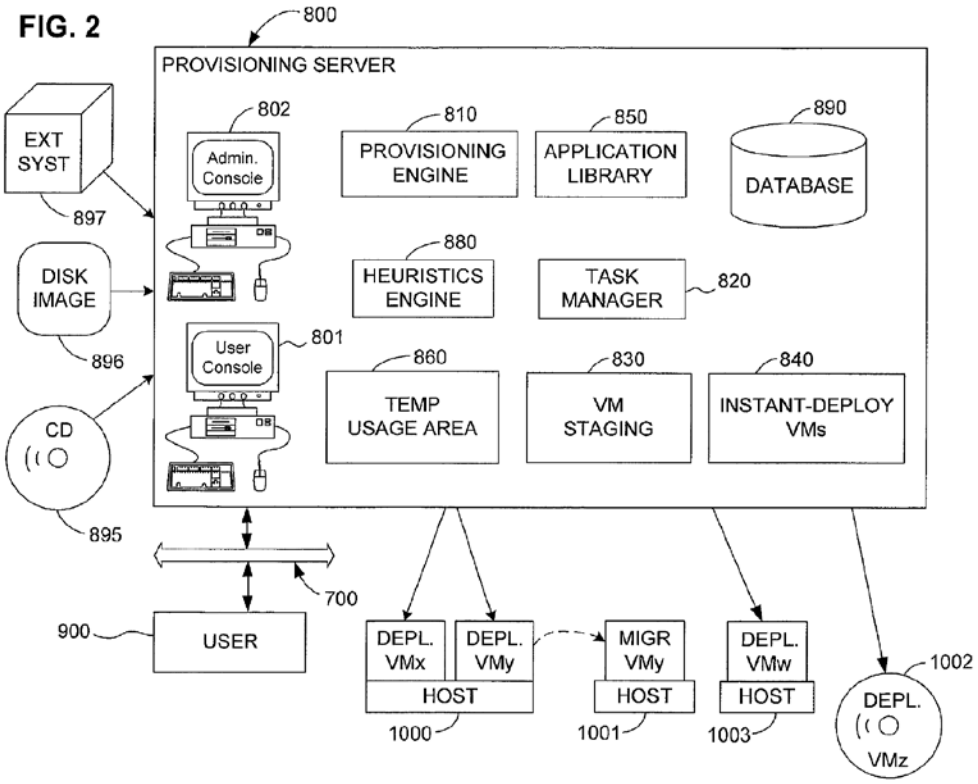
1. Obviousness over Khandekar's Teachings

44. Khandekar renders the challenged claims obvious with its teachings regarding a provisioning server, as explained in the sections below. Khandekar intends for its teachings to be used together, and a POSITA would have found it obvious and been motivated to do so in the context of Grounds 1 and 2. While Khandekar describes aspects of its teachings in terms of "embodiments," a POSITA would have understood, or at least found it obvious, that Khandekar's teachings, as described for Grounds 1 and 2, are meant to be used together because they describe complementary aspects of the invention.

45. For example, Khandekar teaches that "[i]n a preferred embodiment of the invention, a plurality of pre-configured VMs having different configurations are pre-stored" while "[i]n other embodiments of the invention, the requester specifies various components of the desired VM…" *See* Ex. 1004, 4:36-47. The Detailed Description further explains that these pre-stored and custom VMs are meant to be used together in the provisioning server, which allows the user to

choose between pre-stored and custom-built VMs as the first two options. *See, e.g., id.,* 9:10-25 ("The invention provides the user with three choices: 1) Clone an existing, ***pre-built*** model VM … 2) Install an operating system and/or one or more applications … Such a VM is referred to below as a '***custom-built***' VM. 3) Instantly provision a VM by resuming it from a suspended state."). In addition, Khandekar also presents the user with a third choice of instantly-deployable VMs (*see id.*), which are elsewhere described as an alternative embodiment (*see* Ex. 1004, 11:11-35). A POSITA would have been motivated to apply Khandekar's teachings for all three types of VMs because Khandekar teaches using all three VM types together in a provisioning server and offering the user a choice between them.

46. As another example, Khandekar teaches that "most embodiments" include a plurality of physical hosts, and in "***another aspect*** of the preferred embodiment of the invention, the status of the hosts is monitored and the host on which a configured VM is to be deployed is selected heuristically." *Id.,* 4:58-65 (emphasis added). Therefore, Khandekar teaches that automated deployment using its heuristics engine is meant to be used together with VM provisioning described above because it is "another aspect" of the preferred embodiment. *See id.* This is illustrated in Figure 2, which teaches that many aspects of Khandekar's teachings are meant to be used together in the provisioning server. *See id.,* Fig. 2:



47. Figure 2 teaches a provisioning server with, e.g., different types of VMs ("VM Staging" for pre-built and custom VMs, and "Instant-Deploy VMs" for instantly-deployable VMs); an application library; a database; and a "heuristics engine" for deploying images to a plurality of multiple host computers. Therefore, Khandekar teaches that these features are meant to be used together as features of a provisioning server, for example as illustrated in Figure 2, even though they may be labelled as "embodiments" in other parts of the specification. *See* Ex. 1004, Fig. 2, 14:4-18, 16:42-63. The specification describes, for example, that the application library stores applications that are used when creating VM images, which are stored in repositories, and the heuristics engine is used to deploy images from the

repositories to hosts. *See, e.g.*, Ex. 1004, 9:10-25, 16:42-63, 15:40-16:12, 18:56-19:18, 10:27-42, 19:19-20:30. Thus, Khandekar teaches that features of the provisioning server are meant to be used together.

48. Khandekar also teaches the use of VMMs, and a POSITA would have understood, or found it obvious and been motivated in the context of Grounds 1 and 2, to use Khandekar's VMM teachings with the provisioning server because Khandekar's provisioning server and hosts include VMs, which require a VMM to run. A POSITA would have understood this, or found this obvious in the context of Grounds 1 and 2, from basic knowledge of virtualization, which Khandekar explains was "prior art," and also from VMware software, which used VMMs to run VMs. *See supra* §IV; Ex. 1004, 7:7-27, Fig. 1. Furthermore, Khandekar teaches including VMMs with VMs and then providing a VMM together with a VM to a host computer. *See* Ex. 1004, 8:36-44, 13:17-35; *infra* §VI.A.3[1b]-[1c]. Since Khandekar teaches storing VM images in the provisioning server (*see infra* §§VI.A.3[1e]), a POSITA would have found it obvious for the provisioning server to also store VMM images so that it can provide the VMM together with a VM to host computers, as taught by Khandekar. *See id.*

49. After explaining the provisioning server and VMs, Khandekar teaches that "[t]he invention described above is easily extensible to provisioning more than one computer at a time…" Ex. 1004, 23:1-20. "An example of where this would be

useful is where a developer wants to test his web application, which uses a web server on one computer, an application server on another, and a database server on yet another." *Id.* A POSITA would have found it obvious and been motivated to apply these teachings to the provisioning server because Khandekar expressly teaches it, and because web applications were a common use for distributed systems and VMs.

50. A POSITA would have had a reasonable expectation of success when combining Khandekar's teachings because Khandekar's specification and Figure 2 teach their combined use, as explained above.

2. Obviousness and Motivation to Combine Khandekar and Le

51. Khandekar teaches a provisioning server with a repository of VM images and then automatically deploying those VM images on a physical host platform, using a heuristics/rules engine. *See, e.g.*, Ex. 1004, 4:30-35, Abstract. Khandekar teaches monitoring the status of the hosts and then deploying a VM on a host selected based on heuristics/rules. *See id.*, 4:58-65, 10:27-43. Khandekar teaches deploying a VMM along with the VM image. *See, e.g.*, Ex. 1004 8:36-44, 13:17-35 ("a VMM is typically included for each VM in order to act as its software interface with the underlying host system").

52. Le reinforces Khandekar's teachings with implementation details regarding, for example, VMware software, the ratio of VMMs to VMs, and disk

imaging techniques. For example, it was known in the art that a VMM could support multiple VMs. For example, Khandekar teaches that "[i]t would also be possible to use a single VMM to act as the interface to all VMs, although it will in many cases be more difficult to switch between the different contexts of the various VMs …" *See* Ex. 1004, 7:28-38. While Khandekar teaches potential advantages to using one VM per VMM, a POSITA would have also found it obvious to use a single VMM to manage multiple VMs, as explained below.

53. Khandekar was a VMware patent filed in 2002. By 2006, it was common to use multiple VMs per VMM, including for example with VMware ESX software. This is confirmed by a subsequent VMware patent to Le, which teaches a single VMM hosting multiple VMs. *See* Ex. 1005, 65:43-50, FIG. 8. Other products were also available, such as the Xen hypervisor, an x86 VMM that supported multiple VMs. *See* Ex. 1011 at 1, 5, FIG. 1. Additionally, I am aware of companies such as IBM that used multiple VMs per VMM by this time, and a POSITA would have been aware of such usage. Supporting multiple VMs on a single VMM had the known advantage of making it easier to scale a large number of VMs on a single physical server, reducing overhead by reducing the number of concurrent VMM instances on a server, simplifying management of resources since a single VMM can manage all resources, and enabling consolidation of

multiple applications on a single server. *See, e.g.*, Ex. 1008, FIG. 1. The Khanna reference is consistent with my opinion of the state of the art at the time. *See id.*

54. By the time Le was filed in 2003, some VMware products supported hosting multiple VMs on a single VMM, like other products in the state of the art at the time. *See* Ex. 1005, 65:43-50, FIG. 8. VMware software, including its VMM software (ESX), supported this functionality. *See* Ex. 1005, 65:65-66:10, FIG 8.

55. A POSITA would have been motivated to combine Le's teachings regarding the use of multiple VMs per VMM with Khandekar for numerous reasons. First, the Khandekar and Le patents were both assigned to VMware, with overlapping inventors—Khandekar was one of the named inventors for the Le patent, and Le was one of the named inventors on the Khandekar patent. *See* Ex. 1005 at 1; Ex. 1004 at 1. Thus, a POSITA would have understood that Le reflects an updated view of the state of the art for hosting VMs on a VMM in 2003, including updated functionality that was publicly available in VMware's commercial ESX software products. *See* Ex. 1005, 65:43-50, 65:65-66:10, FIG. 8; Ex. 1004, 7:28-38. Thus, a POSITA in 2006 would have been motivated to apply Le's teachings to Khandekar, such that each VMM hosts multiple VMs.

56. Additionally, a POSITA would have been motivated because Le's teachings provide a few advantages here, including reducing overhead of running multiple VMMs on a server, simplifying management of resources on the server

since a single VMM can manage all resources, enabling consolidation of multiple applications on a single server by executing them inside virtual machines, providing support for multiprocessors, and creating images compatible for both physical and virtual machines. *See, e.g.*, Ex. 1005, 46:58-47:14, 18:6-9, 19:12-23; Ex. 1008, FIG. 1. The benefits of virtualization technology for making more effective use of physical servers was well known. Since Khandekar teaches using a single VMM for each VM and suggests it would also be possible to use a single VMM to act as the interface to all VMs, a POSITA would have been motivated to seek other teachings to improve the efficiency of the teachings in Khandekar. This would have led them to Le's teachings of a VMM that runs multiple VMs, which eliminates the need to run multiple VMMs on a server and reduces software overheads.

57. In addition to the use of multiple VMs per VMM, Le further teaches implementation details of image creation and management for distributed computing and virtual machine systems, which a POSITA would have been motivated to combine with Khandekar's teachings because the teachings represent basic implementation details that naturally fit Khandekar's teachings regarding the operation of its provisioning server. In view of Khandekar's teachings, a POSITA would have naturally looked to other patents by the same company and inventors for additional details for implementing and improving upon Khandekar's

teachings. In light of the overlap of inventors (Dilip Khandekar and Bich Le) and the fact that these are both VMware patents directed to features of virtual machine systems, a POSITA would have been motivated to combine teachings from Le with Khandekar.

58. For example, Khandekar teaches that the provisioning server may provide a VMM along with a VM to a host computer. *See, e.g.*, Ex. 1004, 13:17-35. Le teaches imaging techniques that a POSITA would have been motivated to use for this purpose. A VMM is a program running on a physical computer, either as the operating system (Type 1 hypervisor OS) or an application running on top of a separate operating system (Type 2 hypervisor). *See, e.g.*, Ex. 1004, 7:7-27; Ex. 1005, 66:2-10. Le teaches techniques for physically imaging the hard disk of a remote computer, to deploy a cloned image with an operating system (which would include a Type 1 VMM) and applications (which would include a Type 2 VMM) installed. *See, e.g.*, Ex. 1005, 2:19-21, 15:63-65, 16:29-30, 54:14-21, 1:64-3:13; *infra* §VI.A.3[1c], [1f]. Le therefore teaches a mechanism for deploying VMMs to remote computers, which is useful in a variety of common scenarios in distributed computing including (a) installing a VMM onto a computer that does not currently support virtual machines, (b) initializing a new host that is connected to the network, and (c) reimaging a host that is not functioning properly. Le teaches the use of imaging clients to handle the imaging process at the remote computer (*see*

Ex. 1005, Figs. 3-5), and the use of the PXE standard to automatically initialize new computers by installing and then executing the imaging client. *See, e.g.*, Ex. 1005, 37:36-49 ("For example, the PXE (pre-boot execution environment) standard specifies a mechanism for **automatically initializing a new, bare-metal computer when it first appears on the network**. … the server can download an arbitrary program over the network into the computer's memory. The computer executes the program once it is downloaded.") (emphasis added). Therefore, Le teaches techniques for implementing Khandekar's teachings of providing a VMM to host computers, and a POSITA would have found it obvious and been motivated to apply those teachings to Khandekar.

59. Furthermore, Khandekar provides express motivation to apply a solution for determining host configuration, and Le teaches such a solution. Khandekar teaches that the central provisioning server may provide a VMM to a host "as long as the characteristics of the host on which the VM is to be installed are known." *See* Ex. 1004, 13:17-35. Le teaches that the imaging client used to copy/clone a disk (*see, e.g.*, Ex. 1005, Figs. 3-5) includes additional functionality that allows a central repository to determine the hardware characteristics of the remote computer. *See* Ex. 1005, 70:13-35 ("… The registration process causes the imaging client 1021 running from the secondary stack 4100 to ***analyze the computer's hardware configuration and transmit it over the network*** 3000 to the

UCMS server 2101….") (emphasis added). When a new computer (such as a host) is registered, its hardware configuration is analyzed and provided to the central server. *See id.* Therefore, Le's disk imaging teachings address two functions raised by Khandekar: they allow a VMM to be provided from the provisioning server to hosts, and they allow the provisioning server to know the characteristics of the host, so that an appropriate VMM image can be provided. *See id.* These synergies would have motivated a POSITA to combine Khandekar and Le by using Le's imaging teachings to deliver virtualization infrastructure to hosts, including VMware software such as ESX and GSX software, and Le's teaching of virtual machine monitors/managers to the extent Patent Owner attempts to distinguish between these terms—both were components of VMware software that would have obviously been included in the disk imaging techniques described above.

60. The prior art provides additional teaching, suggestion, and motivation to combine Khandekar with Le as described above, using VMM images to provide a VMM to host computers. For example, several prior art references teach loading a VMM (hypervisor) using an image of a VMM. *See* Ex. 1014 ¶[0062] ("The hypervisor also may be loaded first or part of an OS-virtual machine manager image. It may be essentially part of a firmware image."); Ex. 1016 ¶¶[0049], [0043] ("The subsequent steps in the installation procedure of a fixed system remain essentially the same regardless of how the system is actually managed. …

The bootable core and the virtual machine manager image must be installed in a secure partition.").

61. While Le discusses various challenges and solutions for the details of disk imaging, they would not detract from the obviousness or motivation of combining Khandekar and Le because prior art disk imaging solutions were used in the art despite their alleged limitations, and in any case, Le teaches solutions to the alleged limitations. *See, e.g.*, Ex. 1005, 17:6-11 ("The prior art thus provides several different solutions to the problem of creating disk images, all of which suffer from limitations of their own. What is needed is a solution that makes disk imaging possible but without these limitations, or least such that they are less acute. This invention provides such a solution."). Thus, a POSITA would have been motivated to apply prior art imaging solutions taught by Le, as well as the teachings of Le's embodiments to Khandekar.

62. As another example, Le teaches features of commercially available VMM software such as VMware ESX, GSX, and Connectrix VMM software. *See, e.g.*, Ex. 1005, 65:65-66:10. This includes virtual machine monitors and managers of VMware software such as the ESX Server. *See, e.g., id.*, 65:58-64, 66:11-19. A POSITA would have been motivated to use VMware ESX software to implement Khandekar's teachings because Khandekar is a VMware patent, and its VMM teachings would obviously have applied to ESX and other VMware software. *See*

Ex. 1004, 13:17-35 ("Several products are available from VMware, Inc. of Palo Alto, Calif., that configure conventional computers, both stand-alone desktop computers and multi-user servers, to load and run multiple VMs."). Moreover, a POSITA would have been motivated to apply known features of VMware software and systems, including ESX server and Le's teachings of virtual machine monitors and managers, to Khandekar's VMM teachings. *See* Ex. 1005, 13:22-30. Additionally, Khandekar teaches that VM images are created (*see, e.g.,* Ex. 1004, 9:10-25, FIG. 7) and a POSITA would have naturally looked to Le for additional details on how to create images and how to use them to deploy software, including VMware software such as ESX, GSX, and related components, to remote hosts. Lastly, Le teaches the use of disks to store images. *See, e.g., id.,* 60:64-61:8. Hard disks were commonly used to store images. Le further teaches techniques for converting between physical and virtual images. *See, e.g.*, Ex. 1005, 18:6-9, 44:58-48:4, 48:31-55.

63. Le provides many general teachings regarding virtual machine systems and imaging systems, discussed above, and it also provides "exemplifying embodiments of the invention" including a "Universal Computer Management System (UCMS)." *See* Ex. 1005, 59:9-27. Since the UCMS embodies the principles of Le's invention, it would have been obvious that Le's description of the UCMS elaborates on the virtual machine and imaging teachings described

elsewhere in Le's specification, and a POSITA would have been motivated to combine Le's UCMS teachings with related teachings elsewhere in Le's specification. Furthermore, the UCMS is similar to Khandekar's provisioning server because it manages and deploys images from a central repository to virtual machines and physical computers. *See, e.g.*, Ex. 1005, 65:20-33 ("The UCMS can manage virtual machines in addition to physical computers, using a common set of image files. … the UCMS can provide its own virtual machine software components, or interface with existing virtual machine products and integrate their virtual machines into the framework of managed computers."). Therefore, a POSITA would have been motivated to apply Le's UCMS teachings related to virtual machines and imaging to Khandekar's provisioning server, which plays an analogous role in distributed computing with similar functionality.

64. A POSITA would have had a reasonable expectation of success when combining the teachings of Khandekar and Le because the references contain compatible and complementary teachings. Both references are VMware patents directed to features of virtual machine systems, with overlapping inventors. Furthermore, Khandekar teaches the use of a VMM, and Le teaches particular VMM software provided by the same company VMware. Thus, the combination involves a simple substitution of the general VMM taught in Khandekar for known VMM software from the same company and related usage of that software, as

taught by Le. A POSITA would have had a reasonable expectation of success in combining Khandekar and Le given the similarities in their teachings, which stems from the fact that both references are directed to complementary aspects of virtualization and distributed systems, including for example Le's teachings regarding disk imaging techniques, technologies, such as physical and virtual disk imaging, which were well-known by 2006, with remote imaging being commonplace in distributed computing, and it would have been well within the level of ordinary skill to apply those techniques and the other teachings of Le described below. *See supra* §IV.

3. Independent Claim 1

65. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 1 obvious.

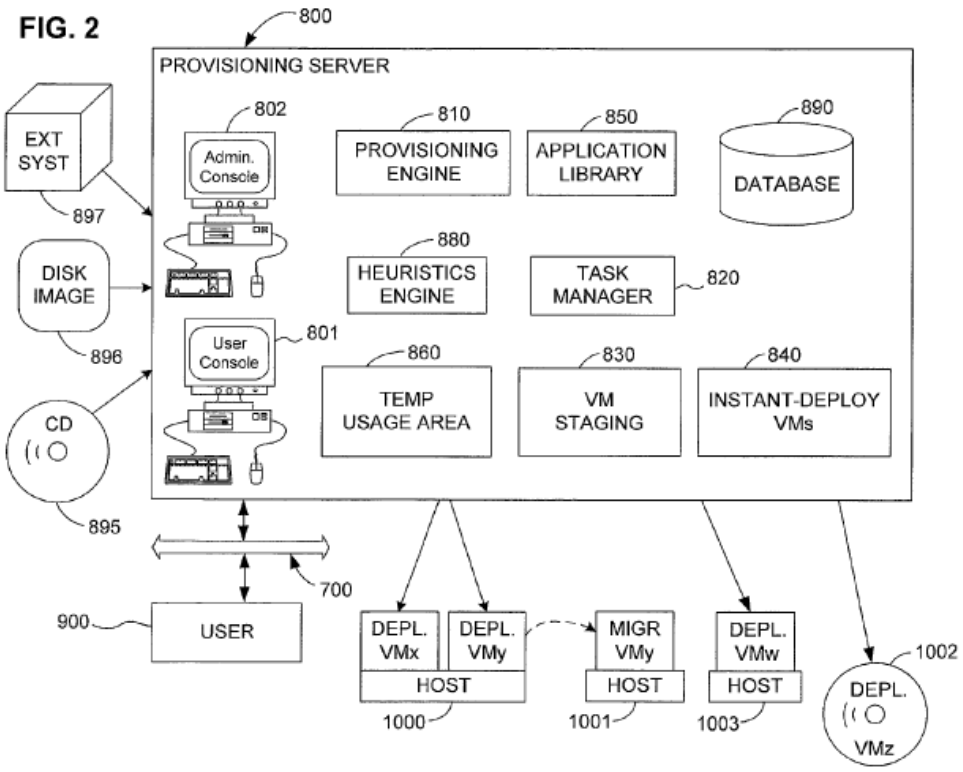
1[a]. A distributed computing system comprising:

66. I understand that a preamble generally does not state a claim limitation. However, to the extent that Patent Owner argues that the preamble states a limitation, it is my opinion that Khandekar teaches or suggests the preamble, as explained below.

67. Khandekar teaches or suggests a distributed computing system. For example, Khandekar teaches a distributed computing system with "a network of cooperating VMs … deployed on respective physical host platforms." Ex. 1004,

5:9-13. “In most embodiments of the invention, there is a plurality of physical hosts on which VMs can be deployed.” *Id.*, 4:58-59. “[A] virtual machine-to-hardware interface, such as a virtual machine monitor (VMM), is installed on each of a plurality of hardware hosts[,]” one of which “is then selected as the host for actual VM deployment.” Ex. 1004, 4:66-5:8. “In the preferred embodiment of the invention, … hosting is preferably provided using a bank or ‘farm’ of servers, each forming one of the hosts.” Ex. 1004 12:40-50.

68. Khandekar teaches “[p]rovisioning a Network of Computers … where the various computers have applications that are configured to interact with applications on other computers”—i.e., a distributed computing system. Ex. 1004 23:1-20, Abstract (“… monitoring the status of VMs and hosts … creating a network of VMs.”), 4:30-35 (“The invention includes a method and system implementation for creating a virtualized computer system based on input information identifying a desired computer configuration.”), 23:26-51 (“FIG. 7 illustrates a method 900 of creating a virtualized computer system.”), 23:54-24:27, FIG. 7. Khandekar teaches a provisioning server within the distributed computing system. *See, e.g., id.*, FIG. 2:



69. A POSITA would have found it obvious to use a distributed computing system, as taught by Khandekar, because virtual machines were primarily used in networked computer settings. *See supra* §I. Khandekar teaches the use of virtual machines, and a central purpose behind virtual machines was to allow easy deployment of VM applications in a network of servers. *See id.* Therefore, Khandekar renders element [1a] obvious. *See* §VI.A.1.

Ground 2: Le's teachings, when combined with Khandekar, further render this claim element obvious.

70. To the extent Patent Owner argues this element is not taught by Khandekar, Le provides additional teachings that complement and elaborate on

Khandekar's teachings, further rendering this limitation obvious. *See* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le).

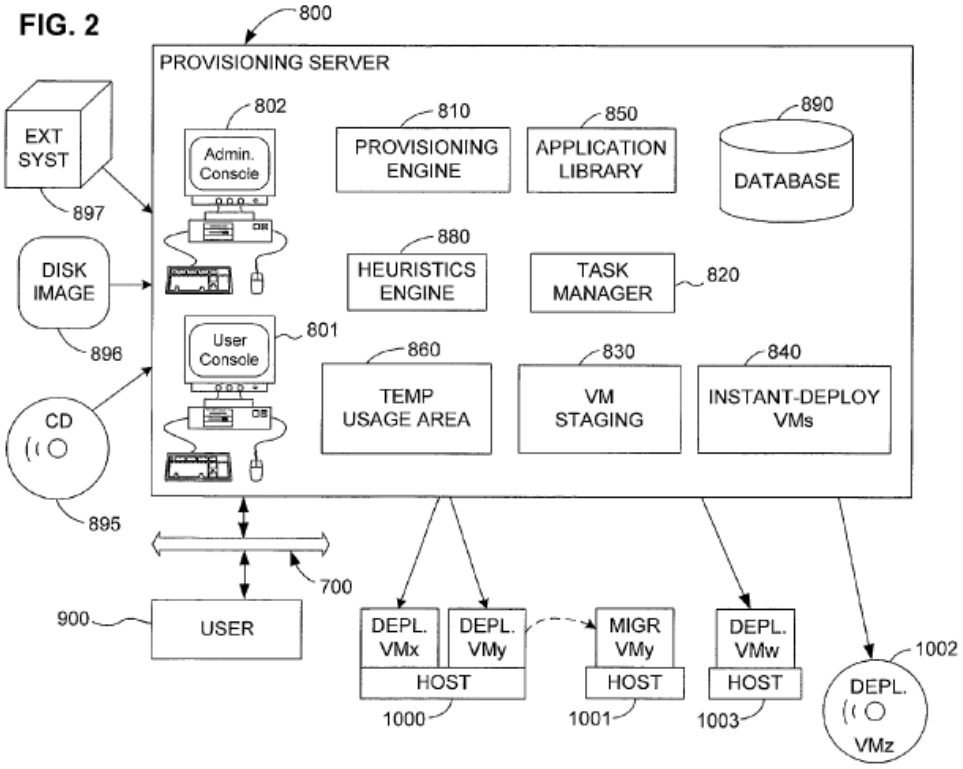
71. For example, Le teaches a distributed computing system, including a UCMS that plays an analogous role to Khandekar's provisioning server, as explained above. *See, e.g.*, Ex. 1005, 59:29-36 ("The UCMS manages a set of physical computers and virtual machines residing on a network. … a virtual machine must reside on a physical computer, equipped with virtual machine software; this computer is called a virtual machine host …"), 53:45-57 ("Several existing configuration management systems allow an administrator to monitor and manage multiple computers running on a network."), 60:19-28 ("computers that are managed by the UCMS … include physical computers, virtual machine hosts …").

72. A POSITA would have found it obvious to combine Le's teachings with Khandekar's teachings because both VMware patents are directed to features of virtual machine systems, as explained above.

73. Therefore, the combination of Khandekar with Le's additional teachings further renders element [1a] obvious. *See* §VI.A.2.

[1b]	**a software image repository comprising non-transitory, computer-readable media operable to store:**

74. Khandekar teaches or suggests a software image repository comprising non-transitory, computer-readable media operable to store the image instances discussed for limitation [1c] and [1e] below. *See infra* §§VI.A.3[1c]-[1e]. For example, Khandekar teaches or suggests a "provisioning server" that acts as a software image repository to store various images. *See, e.g.*, Ex. 1004, 4:36-41 ("… a plurality of pre-configured VMs having different configurations are pre-stored."), 11:36-49 ("applications may be stored in respective libraries …, one such library could contain the code of different operating systems …, as well as other system software such as drivers, etc."), 23:28-51, 23:54-24:27, FIG. 2:



75. The provisioning server includes several libraries/repositories for software images, including library 830 (VM staging repository), *see* Ex. 1004, 16:43-63, 18:23; library 840 (instantly-deployable VM repository), *see* Ex. 1004, 11:21-25, 18:14-15; and library 850 (applications library), *see* Ex. 1004, 18:56-19:2. Staged or pre-configured VMs are stored in image repositories 830 and 840. Ex. 1004, 16:43-63 ("This VM is then ready to be deployed and is stored as one of the staged VMs 830."), 11:21-25 ("… fully configured VMs … are stored in a library 840 of instant deployment VMs[2] …"). A POSITA would have found it obvious to store the VMM images in the provisioning server because Khandekar teaches storing VMs in the provisioning server, and a corresponding VMM is included and installed together for each VM. *See infra* §VI.A.3[1c]; Ex. 1004, 8:36-44 ("… when a VM is installed on a host, then a corresponding VMM is either already installed on the host, or is included and installed together with the VM."), 13:17-35 ("a VMM is typically included for each VM in order to act as its software interface with the underlying host system. … a corresponding VMM to be included along with each staged VM…"). Software and applications are stored in the applications library 850. *See infra* §VI.A.3[1e]; Ex. 1004, 18:56-19:2 ("The

[2] Khandekar refers to "instantly deployable" VMs alternatively as "instant-deploy" VMs, "instant deployment" VMs, "instantly-deployable" VMs, and instantly-provisioned VMs.

applications library 850 is a repository of the software and data needed to install an application 856 in a VM."), 11:50-58 ("… library 850 … contain[s] code for any collection of the myriad of existing applications, from web servers, database servers, spreadsheets and word-processing programs, to games and browsers."), FIG.2. Therefore, Khandekar renders the use of a single repository for images of VMs, VMMs, and software applications obvious.

76. The provisioning server acts as a "central repository" with images of virtual machines and applications, including a database of related information. *See* Ex. 1004, 18:1-42:

> Database 890
>
> The database 890 (see FIG. 2) is a central repository of data required by the various components of the provisioning server. In the preferred embodiment of the invention, this database includes the following information:
>
> 1) Instantly deployable VMs: This is a data structure indexed by the name of the VM configuration, and contains the following fields:
>
> a) Name of the configuration (primary key)
>
> b) Operating system
>
> c) Service pack level for the OS
>
> d) List of applications available
>
> e) List of users configured with accounts on the machine

f) Location of the VM in the instantly-deployable VMs repository 840.

2) Staging VMs: This data structure is required for provisioning pre-built or custom-built VMs and contains the following fields:

a) Operating system name (primary key)

b) Service pack level for the operating system

c) List of applications available in the VM

d) List of users configured with accounts on this VM

e) Location of the VM in the VM staging repository 830.

3) Applications: This data structure keeps information about the applications available for installation in a custom-built VM and has, for example, the following fields:

a) Name of the application (primary key)

b) List of operating systems it runs on

c) Location of the application software in the application library 850

d) Information on hardware properties it needs, such as minimum CPU, memory, disk space, etc.

77. Khandekar teaches that the provisioning server stores image instances but does not limit its teachings to any particular type of storage device. Instead, Khandekar teaches the use of standard computer components, which a POSITA would have understood to include hard disks. *See, e.g.*, Ex. 1004, 9:26-32 ("The

main feature of the invention is a VM provisioning server 800, which, like any conventional computer system, will include system hardware, system software, devices, etc. as needed; these are standard components of any computer and are therefore not described in further detail."). In the 2006 timeframe, a POSITA would have found it obvious based on Khandekar's teachings to store image instances using hard disks, and thus a non-transitory medium. Magnetic hard drives were the predominant form of non-volatile storage in 2006, particularly for storing data in the ranges appropriate for VMMs and software applications. For example, these were often in the GB ranges, and thus it would not have been practical to attempt to store pluralities of such images in DRAM. Large-volume hard disks were ideally situated to store the images taught by Khandekar, for cost, size, and also because they retain their state when the system is reset or powered down, preventing data loss. Having used virtualization products and virtual machines from VMware and Linux for my research at that time, I have direct personal knowledge of the widespread use of hard disks for storing VMMs, VMs, and software applications during that time. Therefore, a POSITA would have found it obvious to store Khandekar's images, as discussed for limitations [1c]-[1e], in non-transitory, computer-readable media such as hard drives. Thus, for the reasons explained above, Khandekar renders this limitation obvious. *See* §VI.A.1.

Ground 2: Le's teachings, when combined with Khandekar, further render this limitation obvious

78. To the extent Patent Owner argues this element is not taught by Khandekar, Le provides additional teachings that complement and elaborate on Khandekar's teachings, further rendering this limitation obvious. *See* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le).

79. For example, Le teaches a software image repository (e.g., server computer 2000 that includes server disk 2010 and template images 4020) comprising non-transitory, computer-readable media operable to store the image instances (e.g., image file 2015, template images 4020) discussed for limitation [1c] and [1e] below. *See infra* §§VI.A.3[1c]-[1e]; Ex. 1005, 25:34-41 ("… image 2015, i.e., the virtual disk"), 26:6-22 ("Finally, the imaging server 2101 can populate the image 2015 by copying files and directories from the source file system(s) to their corresponding destination file system(s)."), 26:63-27:9 (during deployment "the destination computer … load[s] the new operating system deployed from the image 2015."), 66:66-67:8 ("… create template image 4020 from a virtual machine's virtual disk."); 67:25-29 ("… the source virtual machine's virtual disk file, becomes a template image 4020"), FIGS. 4-6:

Fig. 4

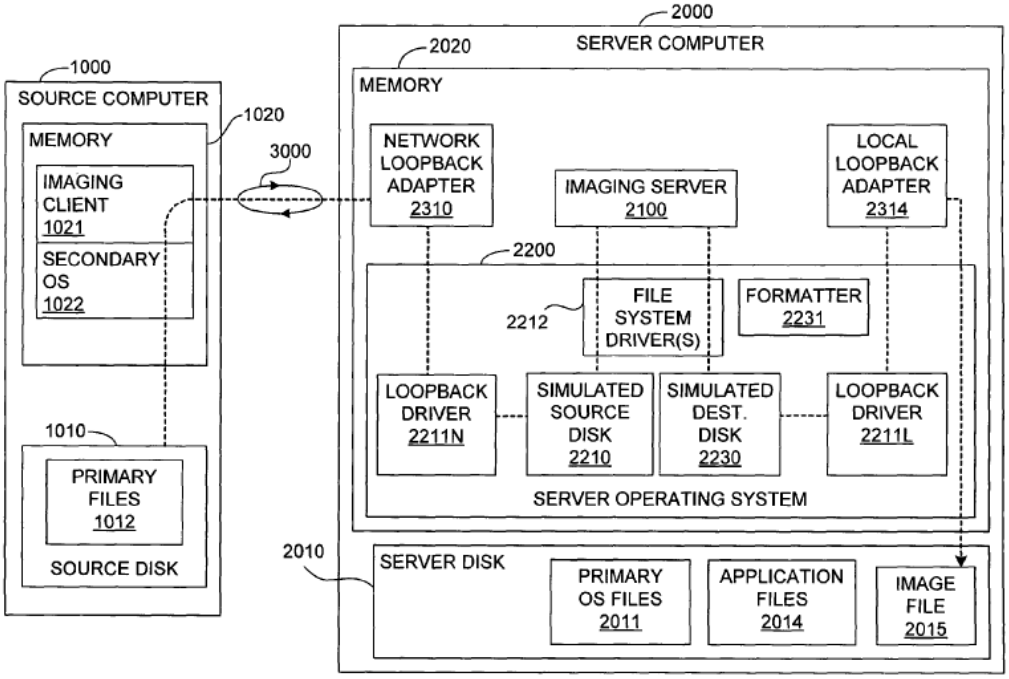
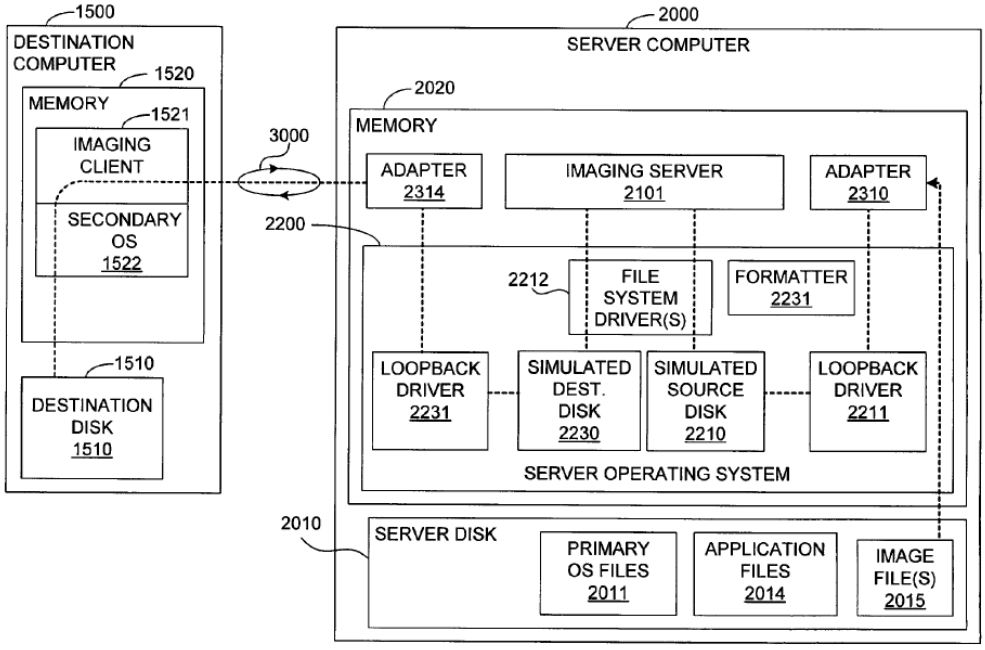
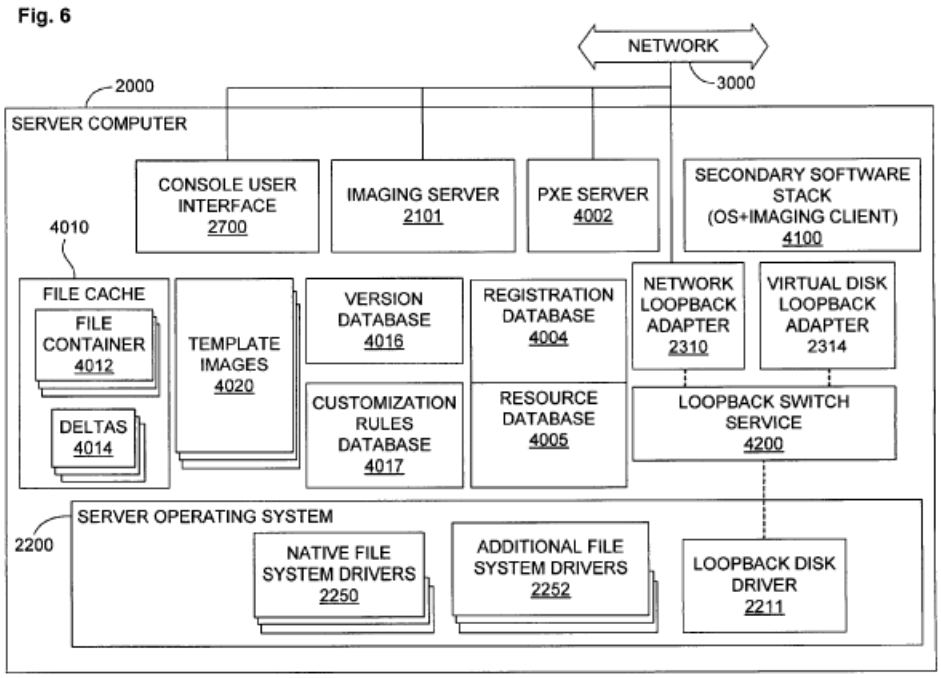

2000
SERVER COMPUTER
2020
MEMORY
1000
SOURCE COMPUTER
1020
3000
MEMORY
IMAGING CLIENT 1021
SECONDARY OS 1022
NETWORK LOOPBACK ADAPTER 2310
IMAGING SERVER 2100
LOCAL LOOPBACK ADAPTER 2314
2200
2212
FILE SYSTEM DRIVER(S)
FORMATTER 2231
LOOPBACK DRIVER 2211N
SIMULATED SOURCE DISK 2210
SIMULATED DEST. DISK 2230
LOOPBACK DRIVER 2211L
SERVER OPERATING SYSTEM
1010
PRIMARY FILES 1012
SOURCE DISK
2010
SERVER DISK
PRIMARY OS FILES 2011
APPLICATION FILES 2014
IMAGE FILE 2015

Fig. 5


1500
DESTINATION COMPUTER
1520
MEMORY
1521
IMAGING CLIENT
SECONDARY OS 1522
3000
2000
SERVER COMPUTER
2020
MEMORY
ADAPTER 2314
IMAGING SERVER 2101
ADAPTER 2310
2200
2212
FILE SYSTEM DRIVER(S)
FORMATTER 2231
LOOPBACK DRIVER 2231
SIMULATED DEST. DISK 2230
SIMULATED SOURCE DISK 2210
LOOPBACK DRIVER 2211
SERVER OPERATING SYSTEM
1510
DESTINATION DISK 1510
2010
SERVER DISK
PRIMARY OS FILES 2011
APPLICATION FILES 2014
IMAGE FILE(S) 2015



A POSITA would have found it obvious that the server disk 2010 is a non-transitory computer-readable medium such as a hard disk because Le teaches that the server disk 2010 is the computer's primary disk and the server computer loads from the disk 2010 at boot up. *See* Ex. 1005, 26:38-51 ("… instructions and data … may be stored on the server computer's primary disk 2010 and loaded into the memory 2020 for execution when needed. … operating system, whose primary files 2011 are also normally stored on the server disk 2010, along with other application files 2014, and are loaded into memory when the server computer is booted."). Le also refers to a computer's primary hard disk as the primary disk. *See* Ex. 1005, 63:20-34 ("… attempt to boot from the primary hard disk. This configuration allows physical computers to boot from their primary disk under

normal conditions"). A POSITA would also have found it obvious that the template images 4020 are stored in a non-transitory computer-readable medium such as hard disk because Le teaches that "template images 4020 can be stored in one or multiple locations," which "include a local disk or a networked file server." Ex. 1005, 60:64-61:8.

80. A POSITA would have found it obvious to combine Le's teachings with Khandekar's teachings because both VMware patents are directed to features of virtual machine systems, as explained above. Therefore, the combination of Khandekar with Le's additional teachings further renders limitation [1b] obvious. *See* §VI.A.2.

[1c]	**(i) a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes,**

81. Khandekar teaches or suggests this limitation. For example, Khandekar teaches or suggests a software image repository comprising non-transitory, computer-readable media operable (*see supra* §VI.A.3[1b]) to store a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes. As an initial matter, Khandekar teaches a virtual machine manager (e.g., "VMM") that is executable on a plurality of application nodes (e.g., hosts). *See, e.g.*, Ex. 1004, 4:66-5:8 ("… a virtual machine-to-hardware interface, such as a virtual machine monitor (VMM), is installed on each

of a plurality of hardware hosts."). "A VMM is usually a thin piece of software that runs directly on top of the host and virtualizes all, or at least some of the resources of the machine, or at least of some machine." *Id.*, 7:7-27. "As is mentioned above, a VMM is typically included for each VM in order to act as its software interface with the underlying host system." *Id.*, 13:17-35. "Several products are available from VMware, Inc. of Palo Alto, Calif., that configure conventional computers, both stand-alone desktop computers and multi-user servers, to load and run multiple VMs." *Id.* A POSITA would have found it obvious that Khandekar's teaching of a "virtual machine monitor" satisfies the claimed "virtual machine manager" because the references use their respective terms to refer to the same functionality, as explained below. *See infra* ¶¶88-94 (Subsection "Virtual Machine Manager").

82. Khandekar teaches that the software image repository is operable to store a plurality of image instances of VMs, which includes for example "staged VMs." *See supra* §VI.A.3[1b] (discussing VM images); *infra* §VI.A.3[1e]; *see, e.g.*, Ex. 1004, 16:43-63 ("This VM is then ready to be deployed and is stored as one of the staged VMs 830."), 11:21-25, 15:40-16:12, 10:27-42, 23:28-51, FIGS. 2, 3; *supra* §VI.A.3[1b]. Khandekar further teaches including a corresponding VMM with each VM. *See* Ex. 1004, 13:17-35 (emphasis added):

> As is mentioned above, ***a VMM is typically included for each VM in order to act as its software interface with the underlying host system***. In the preferred embodiment of the invention, a VMM is already pre-installed on each host 1000, 1001 so as to ensure compatibility and proper configuration. Several products are available from VMware, Inc. of Palo Alto, Calif., that configure conventional computers, both stand-alone desktop computers and multi-user servers, to load and run multiple VMs. It would also be possible, however, for ***a corresponding VMM to be included along with each staged VM, as long as the characteristics of the host on which the VM is to be installed are known.***

When the provisioning server provides a VM image to a host, it may also include a VMM together with the VM. *See id.*, 8:36-44 ("… when a VM is installed on a host, then a corresponding VMM is either already installed on the host, or is included and installed together with the VM.").

83. Since VM images are stored in the provisioning server, a POSITA would have found it obvious to likewise store a plurality of VMM images in the provisioning server. *See, e.g.*, Ex. 1004, 8:36-44, 13:17-35. A POSITA would have found it obvious for the software image repository to be operable to store a plurality of image instances of a virtual machine manager (e.g., VMM) so that VMM images may be deployed with the VM images stored in the repository, as taught by Khandekar. *See id.* A POSITA would have found it obvious to follow these teachings from Khandekar, with VMM images being stored in the

provisioning server and provided to hosts together with VM images, because doing so would have allowed the hosts to be remotely initialized with VMM software and configured to support virtual machines, thereby facilitating one of Khandekar's goals: automated, flexible deployment of virtual machines to physical host platforms. *See, e.g.*, Ex. 1004, 4:30-35. If a host does not have a VMM and therefore cannot run virtual machines, Khandekar's teachings allow a VMM to be installed onto the host, making the host ready to receive and run virtual machines, without the need for a human administrator to setup the physical machine. Khandekar's teachings thus remove the need to manually install VMM software on physical hosts, which can be advantageous when the number of hosts is large, for example in a server cluster with hundreds or thousands of computers. Furthermore, as explained above, the teachings allow VMs to be assigned to hosts even if the host is not yet configured to support virtual machines, allowing increased flexibility when assigning VMs to hosts.

84. When following Khandekar's teachings, a POSITA would have found it obvious to store a plurality of VMM images to account for the different characteristics of the hosts. Khandekar teaches that a corresponding VMM may be included along with each staged VM, "as long as ***the characteristics of the host*** on which the VM is to be installed are known." Ex. 1004, 13:17-35 (emphasis added). This is because the host computers can have different software and hardware

configurations. *See, e.g.*, Ex. 1004, 8:20-36 ("… any VM may therefore be migrated freely, even to other hardware and software platforms …"), 6:13-22 ("… this invention does not presuppose any particular host operating system …"). Therefore, when a VMM image is provided along with a VM image to a host, it would have been obvious to provide an appropriate VMM image for the characteristics of the host. A POSITA would have found it obvious to store a plurality of VMM images to accommodate the different software and hardware configurations, so that a VMM image can be readily deployed to any host along with the VM. *See* Ex. 1004, 8:36-44, 13:17-35. This was common in the art, for example to create separate images for the different computer types or operating systems that Khandekar teaches for its hosts. Doing so would have allowed the provisioning server to provide a VMM that can run on the particular configuration of a target host, as taught by Khandekar. Moreover, it would have been obvious to include at least two different VMM images, one for Type 1 hypervisors, which act like a special-purpose operating system kernel, and another for Type 2 hypervisors that run on top of another operating system, because Khandekar teaches both options for the VMM. *See, e.g.*, Ex. 1004, 7:7-27. Furthermore, Khandekar teaches including VMMs with VMs and then providing a VMM together with a VM to a host computer. *See* Ex. 1004, 8:36-44, 13:17-35. Since Khandekar teaches that the provisioning server acts as a repository for storing VM images (*see, e.g.*, Ex. 1004,

4:36-41, 11:21-25, 16:43-63), a POSITA would have found it obvious for the provisioning server to also store VMM images so that it can provide the VMM together with a VM to host computers, as taught by Khandekar. *See id.* Indeed, Khandekar teaches that the provisioning server includes a database and several libraries; therefore, Khandekar teaches or suggests that the provisioning server includes storage infrastructure, and a POSITA would have found it obvious and been motivated to store VMM images in the provisioning server, so that the VMM images may be readily deployed to host computers. Thus, it would have been obvious to store a plurality of VMM images to accommodate Khandekar's plurality of host platforms, with different images for different host hardware, operating systems, and host configurations.

85. The combination of Khandekar and Le (Ground 2) provides additional teachings on VMM images, as explained below.

86. **Application Nodes**. The '138 patent explains that "application nodes" are "computing nodes." *See* Ex. 1001, 3:59-64 ("the collection of computing nodes forming distributed computing system 10 are logically grouped within a discovered pool 11, a free pool 13, an allocated tiers 15…"), 4:26-24 ("Control node 12 may dynamically reallocate an unallocated node from free pool 13 to allocated tiers 15 as an application node 14."), FIG. 1:

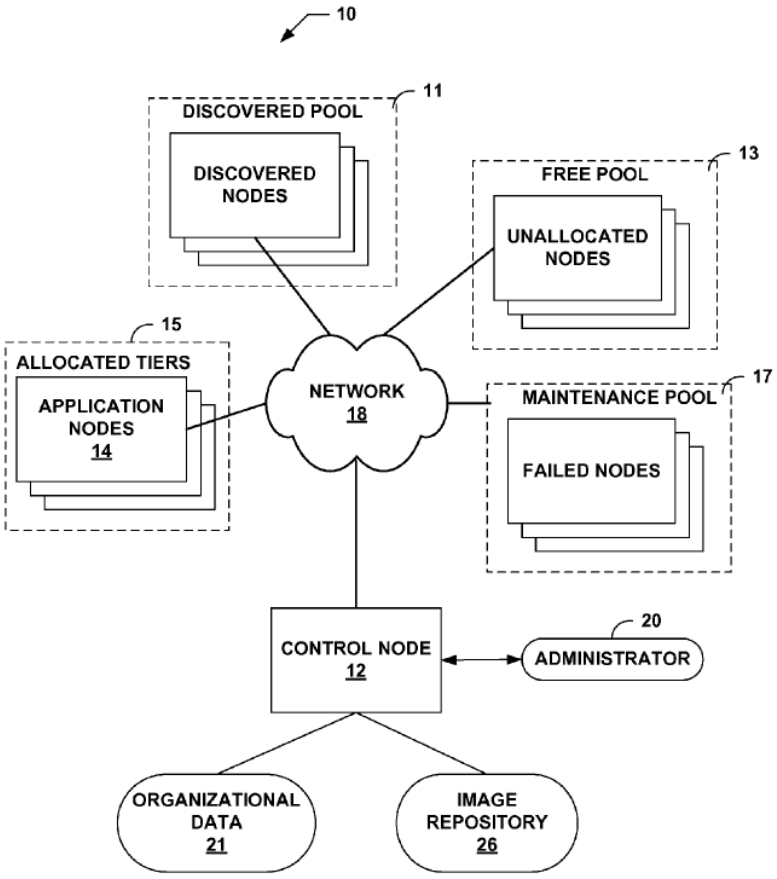


FIG. 1

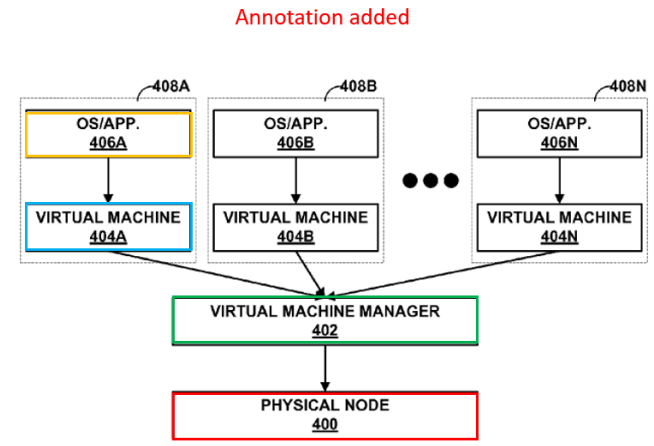
The ’138 patent further explains that “[w]ithin distributed computing system 10, a computing node refers to ***the physical computing device***.” Ex. 1001, 3:65-4:5 (emphasis added), 1:16-25 (“Each of the computing nodes in the distributed computing system is typically a separate, independent computing device interconnected with each of the other computing nodes via a communications medium, e.g., a network.”).

87. Khandekar teaches application nodes in the form of “hosts,” which it also refers to as “host servers” and “host computers.” *See* Ex. 1004, 5:25-36 (“The

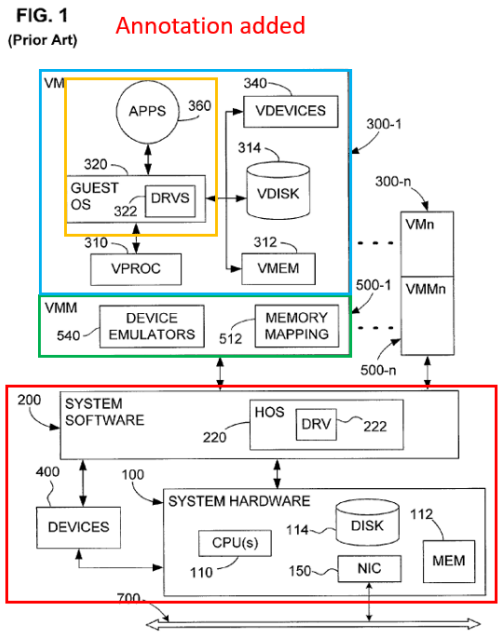
chosen or specified virtual machine is then loaded onto a physical platform such as a host server."), 5:52-58 ("… the hardware platform 100 and the system software 200 thus form a 'host' for the VMs"), 12:40-50 ("In the preferred embodiment of the invention, … hosting is preferably provided using a bank or 'farm' of servers, each forming one of the hosts."), 10:27-43 ("At least one host computer 1000, 1001 is made available to the provisioning server, and the heuristics engine 880 also selects which host machine 1000, 1001 is most appropriate to provision and deploy the VM on."), FIG. 2. A POSITA would have found it obvious that Khandekar's hosts satisfy the claimed the application nodes because the hosts are computing devices or servers that form the computing nodes of the distributed computing system.

88. **Virtual Machine Manager.** A POSITA would have understood Khandekar's teachings of a virtual machine monitor to satisfy the claimed "virtual machine manager" despite the slight difference in terminology. In the prior art, the term "virtual machine manager" was often used to refer to a virtual machine monitor. *See, e.g.,* Ex. 1014 ¶[0029] ("further note that as used herein, virtual machine monitor and virtual machine manager, as well as VMM, can be considered equivalent").

89. Both Khandekar and the ’138 patent use these terms to refer to the same functionality, e.g., the software that manages a VM. *Compare* Ex. 1001, FIG. 24 with Ex. 1004, FIG. 1.



Ex. 1001, FIG. 24



Ex. 1004, FIG. 1

90. The ’138 patent explains that a virtual machine manager is a software program that is capable of managing one or more virtual machines. *See* Ex. 1001, 32:59-33:24:

> As used herein, a virtual machine manager is a software program that is capable of managing one or more virtual machines. For example, virtual machine manager 402 may be an instance of VMWare ESX software produced by VMWare, Inc. of Palo Alto, Calif.

The ’138 patent does not purport to invent a virtual machine manager and instead relies on VMMs that were known in the art, such as “VMWare ESX software.” *See id.*

91. Khandekar teaches that a virtual machine monitor manages the execution of virtual machines. *See, e.g.*, Ex. 1004, 6:62-7:6 (“Some interface is usually required between a VM and the underlying host platform … As is mentioned above, the host platform is responsible for actually executing VM-issued instructions and transferring data to and from the actual, physical memory 112 and storage devices 114. This interface is often referred to as a ‘virtual machine monitor’ (VMM).”). “A VMM is usually a thin piece of software that runs directly on top of the host and virtualizes all, or at least some of, the resources of the machine, or at least of some machine.” *Id.*, 7:7-27. “[T]he interface exported to the respective VM is the same as the hardware interface of the machine, or at least of some predefined hardware platform, so that the guest OS 320 usually cannot determine the presence of the VMM. The VMM also usually tracks and either forwards (to the HOS 220) or itself schedules and handles all requests by its VM for machine resources as well as various faults and interrupts.” *Id.*

92. Furthermore, Khandekar is a VMware patent that teaches the use of VMMs that run directly on the underlying system hardware or alternatively on a Host OS, but in either case, the VMMs manage the execution of the VMs. *See id.*,

6:62-7:27. VMMs had long been known in the art; Khandekar teaches that "[t]he general features of VMMs are known in the art and are therefore not discussed in detail here." *Id.* Given Khandekar's teachings and the fact that it is a VMware patent, a POSITA would have found it obvious to use known VMM software, including VMware ESX software. *See id.*

93. Therefore, a POSITA would have understood, or at least found it obvious, that Khandekar teaches the claimed virtual machine manager. Alternatively, to the extent it is argued that Khandekar's VMM teaching does not satisfy the claimed "virtual machine manager," the additional teachings of Le further render this limitation obvious.

94. For the reasons explained above, Khandekar by itself renders this limitation obvious. *See* §VI.A.1.

Ground 2 (Khandekar in View of Le) Provides an Alternative Basis for the "Virtual Machine Manager"

95. The phrase "virtual machine manager" did not have a universal definition in the art. As explained above, "virtual machine manager" was often used to refer to the functionality of a virtual machine monitor. (*see*, *e.g.*, Ex. 1014 ¶[0029]), which is how the '138 patent uses the term.

96. To the extent it is argued that a virtual machine monitor such as Khandekar's does not satisfy the claimed "virtual machine manager," Le expressly

teaches a virtual machine "manager." *See* §VI.A.2. Le uses the phrase "virtual machine manager" to refer to other aspects of the system, including creating, destroying, and maintaining VM files on the host. *See, e.g.*, Ex. 1005, 66:11-32:

> The virtual machine manager 6200 is typically also responsible for creating, destroying and maintaining virtual machine files residing on the host 6000. It may use a local registration database file 6210 to keep track of the virtual machines hosted on the computer. In order to integrate with the UCMS, the virtual machine monitor 6300 must expose a network interface to the UCMS imaging server 2101 and this interface needs to expose a minimum set of required service functions.

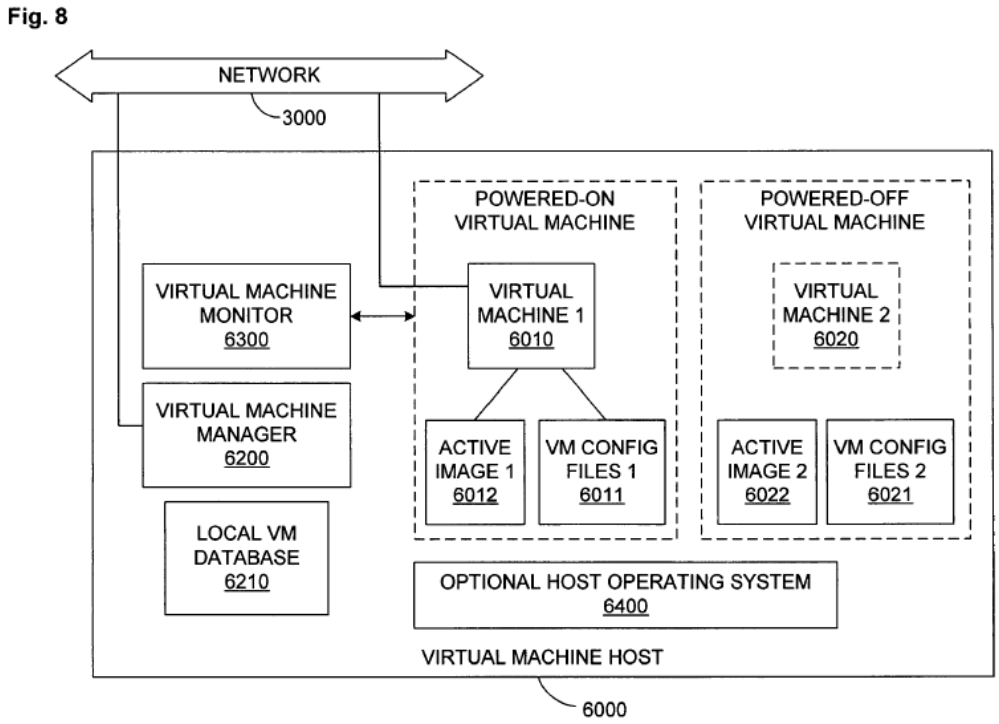
One of the functions "causes the virtual machine monitor to register the new virtual machine on the host." Ex. 1005, 67:20-32. Le's virtual machine monitor and manager may also power on and off the virtual machines. *See* Ex. 1005, 66:48-64 ("Once a virtual machine is successfully deployed from a template image 4020, the virtual machine manager 6200 may optionally power it on (such as virtual machine 6010) …"); 67:9-19 ("… the user (via the imaging server) or the imaging server itself can send a message to the UCMS agent, via the VM manager 6200 and the VMM 6300, to request the guest operating system to shut itself down and power-off the virtual machine hardware …").

97. Like Khandekar, Le teaches that the virtual machine monitor controls and manages virtual machines, including by managing the interactions between software in the virtual machines and the host's physical resources. *See* Ex. 1005,

65:65-66:2 (“When a virtual machine is powered on, a virtual machine monitor program 6300 controls it and manages the interactions between the software—commonly called ‘guest software’—running inside of the virtual machine and the host’s physical resources, such as hardware devices.”). Therefore, Le teaches a virtual machine “monitor” performing the functionality of a virtual machine “manager” as that term is used by the ’138 patent. *See* Ex. 1001, 32:59-67 (“As used herein, a virtual machine manager is a software program that is capable of ***managing one or more virtual machines***. … When deployed on physical node 200, virtual machine manager 402 hosts virtual machines …”) (emphasis added).

98. To the extent it is argued that Khandekar’s teaching of a “virtual machine monitor” does not satisfy the claim limitation of a “virtual machine manager” in the ’138 patent, Le explicitly teaches a virtual machine manager, as explained above. Moreover, Le is a VMware patent that teaches the use of VMware software including “VMware ESX Server” software (*see, e.g.*, Ex. 1005, 65:65-66:10, 13:22-30), which is the same software that the embodiments of the ’138 patent use for a virtual machine manager. *See* Ex. 1001, 32:62-64 (“[V]irtual machine manager 402 may be an instance of ***VMWare ESX software*** produced by VMWare, Inc. of Palo Alto, Calif.”) (emphasis added). The ’138 patent does not purport to invent the virtual machine manager and instead uses the term to refer to the functionality of VMware ESX software, which is taught by Le. Therefore, Le

teaches the claimed "virtual machine manager," and the combination of Khandekar and Le teaches a virtual machine manager that is executable on a plurality of application nodes (e.g., virtual machine host 6000). *See*, *e.g.*, Ex. 1005, FIG. 8:



Ground 2 (Khandekar in View of Le) Provides an Additional Basis for the Plurality of Image Instances of a Virtual Machine Manager

99. Le provides additional disk image teachings that complement and elaborate on Khandekar's teachings, further rendering it obvious for a plurality of VMM images to be stored in the provisioning server. *See supra* §VI.A.2 (explaining how and why Le's disk image teachings would have been combined with Khandekar). As explained above, Khandekar teaches the provisioning server providing VMMs to hosts. Le further teaches various disk-imaging techniques that

facilitate the remote deployment of software onto host machines, techniques which a POSITA would have found obvious to use for deploying VMM images.

100. Le explains that the prior art included techniques for remotely imaging a host computer, including techniques for capturing and deploying images over a network. *See, e.g.*, Ex. 1005, 1:64-3:13 ("Image capture is the process of creating an image file from a computer's disk. … During capture, disk data is transferred from the client to the server over a network or a cable. The reverse of the capture process is called deployment. …"). Le explains that disk imaging can be "used to clone a computer; an image of a first computer can thus be deployed to other computers." *Id.*, 2:19-21.

101. Le teaches "us[ing] disk imaging to quickly provision a bare-metal computer on the network ***with a complete software stack consisting of an operating system and a set of core applications***." Ex. 1005, 54:14-21 (emphasis added). "Disk imaging technology enables backup and rapid deployment of computers. A disk image is an ordinary file that represents a snapshot of a disk's contents." Ex. 1005, 15:63-65, 16:29-30 ("Disk imaging is often used as a cloning technology for rapid computer deployment."). Disk images included the operating system and software applications. *See, e.g.*, Ex. 1005, 8:10-23, 10:55-62 ("Before the initial image is captured, the base computer is configured with an operating system and common set of software applications that are required on all clones.

Any computer deployed from this image would inherit the same set of software."), 22:32-44 ("…A source computer 1000 is equipped with a source disk 1010 containing a set of primary files 1012, including the operating system's files and other files (including application files, data files, etc.)."), 67:31-40.

102. Type 1 VMMs, also referred to as the hypervisor OS, acted as the operating system. *See* Ex. 1004, 7:7-14; Ex. 1005, 13:12-21. Type 2 VMMs ran as software application on top of a separate OS, such as Windows or Linux. *See* Ex. 1004, 7:14-27; Ex. 1005, 13:5-11. Le teaches the use of VMware ESX software, which is a Type 1 VMM, and VMware GSX software, which is a Type 2 VMM. *See, e.g.*, 1005, 66:2-10. Thus, a POSITA would have found it obvious and been motivated to use disk imaging techniques to install a VMM onto a host computer because disk imaging techniques allowed rapid deployment of disk images—including the operating system and software applications such as Type 1 and Type 2 VMMs, respectively—onto remote computers over a network. *See, e.g.*, Ex. 1005, 15:63-65, 16:29-30. As an example, a POSITA would have found it obvious to use images of VMware ESX and GSX software, as explained above, to initialize a host if it does not already have a VMM so that the host can support VMs. As another example, disk imaging techniques can also be used to re-initialize a host that is no longer functioning properly.

103. The use of VMM images in this manner was known in the art. *See, e.g.*, Ex. 1014 ¶[0062] ("The hypervisor also may be loaded first or part of an OS-virtual machine manager image. It may be essentially part of a firmware image."); Ex. 1016 ¶¶[0049], [0043] ("The subsequent steps in the installation procedure of a fixed system remain essentially the same regardless of how the system is actually managed. … The bootable core and the virtual machine manager image must be installed in a secure partition."). Thus, the prior art includes teachings, suggestions, and motivations to combine Khandekar and Le in the manner described above, using a plurality of VMM images to practice Khandekar's teachings.

104. Le further teaches storing a plurality of such images (e.g., image file 2015, template images 4020) in a server computer (e.g., server computer 2000 that includes server disk 2010 and template images 4020), which acts as a central repository analogous to the provisioning server in Khandekar. *See, e.g.*, Ex. 1005, 26:53-62 ("… a stored disk image 2015 (more than one may of course be stored)"), 26:6-22 ("Finally, the imaging server 2101 can populate the image 2015 by copying files and directories from the source file system(s) to their corresponding destination file system(s)."), 26:53-57 ("… the user selects a stored disk image 2015 (more than one may of course be stored) using any conventional method."), 54:14-21 ("A computer management system can use disk imaging to quickly provision a bare-metal computer on the network … A single template image, or a

small set of core template images, is used to create clones."), 66:66-67:8, 67:25-20, FIGS. 4-5, FIG. 6 ("TEMPLATE IMAGES 4020"), FIG. 9, 67:31-40:

> UCMS users and, in particular, a UCMS administrator, decide how many images 4020 to maintain on the UCMS server 2000, what to put in images and what images to deploy onto specific computers. A typical image encapsulates the software stack, as illustrated in FIG. 9. Furthermore, FIG. 9 shows a destination computer whose primary disk 7110 was deployed from the image. The stack 7100 contains an arbitrary operating system 7200, and application software 7111 including one or more software applications 7114 and an arbitrary set of data files (not shown).

105. It would have been obvious, when combining Khandekar and Le, for the provisioning server to store VMM images and provide them to hosts along with VM images for virtual machines. For example, Le teaches an "exemplifying embodiment," the "Universal Computer Management System (UCMS)," (Ex. 1005, 59:21-27), which is analogous to Khandekar's provisioning server. *See supra* §VI.A.2. The UCMS manages and deploys images to virtual machines and physical computers. *See, e.g.*, Ex. 1005, 65:20-33.

106. A POSITA would have found it obvious and been motivated to follow Le's teachings because storing a plurality of images would account for different computer hardware and software platforms for the host computers, as explained above. Khandekar teaches consideration of host characteristics when providing a

VMM. *See* Ex. 1004, 13:17-35, 8:20-36, 6:13-22. Le reinforces the obviousness of this approach, explaining that "[a] common but inelegant solution to the hardware compatibility issue is to create one image file per family of similar computers." *See, e.g.*, Ex. 1005, 9:66-10:9, 16:29-41. Thus, Le teaches that creating different images for each family of computer hardware was common in the art despite being potentially inelegant or having potential drawbacks. *See id.* Real-world engineering solutions typically include advantages and drawbacks, and nothing described in Le would have prevented a POSITA from utilizing known imaging techniques that were used in the art.

107. Additionally, similar to Khandekar, Le teaches multiple VMware software options for the VMM including a Type 2 hypervisor that runs on a host OS (GSX software) and a Type 1 hypervisor that runs directly on the hardware (ESX software). *See* Ex. 1005, 66:2-10 ("VMware GSX Server and Connectix Virtual Server, access the host's physical resources through the services of an industry-standard host operating system 6400 … other products, like VMware ESX Server, include a virtual machine monitor 6300 and system-level kernel capable of managing physical resources directly without the need for a host operating system."). Thus, it would have been obvious for the provisioning server to store at least two VMM images, for GSX and ESX software, which is a plurality (i.e., two or more) of VMM images. Furthermore, as explained above, Khandekar teaches

consideration of the characteristics of hosts when providing a VMM, and thus a POSITA would have found it obvious to store multiple VMM images, for both GSX and ESX software, to account for different software and hardware configurations of the hosts. *See, e.g.*, Ex. 1004, 13:17-35, 8:20-36, 6:13-22.

108. Le teaches extensive details regarding the capture and deployment of images. The ’138 patent does not recite claim limitations directed to such lower-level details, such as challenges and solutions to handling different disk and image file formats. Instead, claim 1 recites higher-level limitations directed to storing images in a repository and copying images to application nodes. The ’138 patent relies on a POSITA’s knowledge regarding mechanisms for the more granular details of image capture and deployment. Those details are not required by the claims, but they are, nonetheless, taught by Le.

109. While Le discusses various challenges and solutions for the details of disk imaging, those details are beyond the scope of the challenged claims. Nor would those challenges detract from the obviousness or motivation of combining Khandekar and Le because prior art disk imaging solutions were used in the art despite their alleged limitations, and in any case, Le teaches solutions to the alleged limitations. *See, e.g.*, Ex. 1005, 17:6-11 (“The prior art thus provides several different solutions to the problem of creating disk images, all of which suffer from limitations of their own. What is needed is a solution that makes disk

imaging possible but without these limitations, or least such that they are less acute. This invention provides such a solution."); *supra* §VI.A.2.

110. A POSITA would have recognized that Le does not does not present insurmountable or prohibitive problems to using prior art solutions—at most, Le teaches limitations or difficulties in the prior art, typical of real-world engineering solutions, that it seeks to improve. While a POSITA would have been motivated to apply Le's invention, the publication of the Le patent would not have suddenly foreclosed a POSITA from using the disk imaging techniques that were already known and used in the art despite their alleged difficulties, and a POSITA would also have been motivated to use prior art solutions to fit the needs of particular projects.

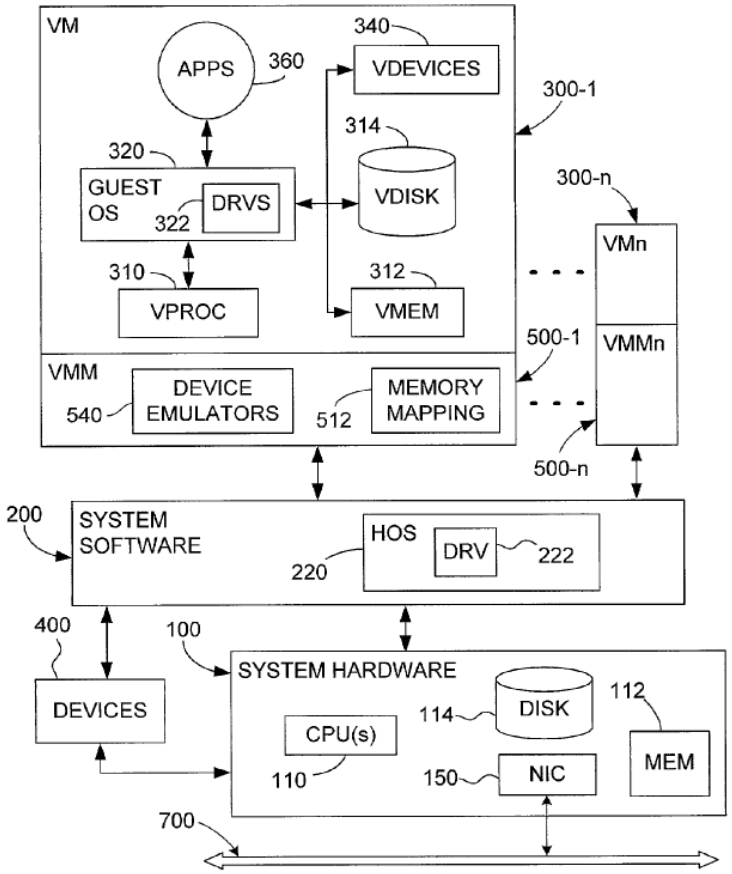
111. Therefore, the combination of Khandekar with Le's additional teachings further renders limitation [1c] obvious. *See* §VI.A.2.

[1d]	**wherein when executed on the applications nodes, the image instances of the virtual machine manager provide a plurality of virtual machines, each of the plurality of virtual machine operable to provide an environment that emulates a computer platform, and**

112. Khandekar teaches or suggests this limitation. For example, Khandekar teaches or suggests that, when executed on the applications nodes (e.g., hosts), the image instances of the virtual machine manager (e.g., VMM) provide a plurality of virtual machines. For example, Khandekar Figure 1 teaches that the

image instances of the VMM (labelled 500-1 “VMM” through 500-n “VMMn”) are executed on the host computer and provide a plurality of virtual machines (labelled 300-1 “VM” through 300-n “VMn”).

FIG. 1
(Prior Art)



113. “FIG. 1 shows the main components of a computer system that **includes one or more virtual machines (VMs)**. The illustrated system includes an underlying system hardware platform 100, system software 200, and **a plurality of virtual machines (VMs) 300-1, …, 300-n** that run on the system software 200; the

hardware platform 100 and the system software 200 thus form a 'host' for the VMs." Ex. 1004, 5:51-58 (emphasis added). The image instances of the VMM are executed on the host computer, allowing it to run a plurality of VMs. *See* Ex. 1004, 13:17-35 (emphasis added):

> As is mentioned above, a VMM is typically included for each VM in order to act as its software interface with the underlying host system. … Several products are available from VMware, Inc. of Palo Alto, Calif., that configure conventional computers, both stand-alone desktop computers and multi-user servers, **to load and run multiple VMs.** …

Ex. 1004, 7:7-27 ("A VMM is usually a thin piece of software that runs directly on top of the host and virtualizes all, or at least some of the resources of the machine, or at least of some machine.").

114. To the extent Patent Owner contends that this claim limitation [1d] requires that each image instance of a VMM provides a plurality of virtual machines, Khandekar teaches this as well. For example, Khandekar teaches that commercial VMware products allow desktop computers and multi-user servers "to load and run multiple VMs." Ex. 1004, 13:17-35. And Khandekar teaches a single VMM providing the interface for multiple VMs. *See* Ex. 1004, 7:28-38:

> It would also be possible to use a single VMM to act as the interface to all VMs, although it will in many cases be more difficult to switch between the different contexts of the various VMs (for example, if

> different VMs use different guest operating systems) than it is simply to include a separate VMM for each VM. The important point is simply that some well-defined, known interface should be provided between each VM and the underlying system hardware 100 and software 200.

When multiple VMs are used with a single VMM, Khandekar notes that it may be more difficult to switch between different VM contexts if different VMs use different guest operating systems. *See id.* Nonetheless, despite possible difficulties in 2002, it was common by 2006 to use multiple VMs for each VMM because of its known advantages and support in commercial software such as VMware software. *See supra* ¶¶52-54. By 2006, therefore, a POSITA would have been motivated and found it obvious to apply Khandekar's teachings where each image instance of a VMM provides a plurality of virtual machines.

115. Khandekar further teaches or suggests each of the plurality of virtual machines (VMs) is operable to provide an environment that emulates a computer platform. *See, e.g.*, Ex. 1004, 7:39-47 ("… the VM is structured and performs as a complete 'real' computer system, …"), 6:23-43 (emphasis added):

> As is well known in computer science, a virtual machine (VM), which is sometimes also referred to as a "virtual computer," is a software abstraction—a "virtualization"—of an actual physical computer system. Actual execution of instructions for the VM is ultimately carried out on the system hardware 100, so that the VM may be

> considered to be a "guest" system running on the "host" platform, which includes the system hardware and software 100, 200. Like physical computers, each VM will typically include one or more virtual CPUs 310 (VPROC), a guest operating system 320 (which may, but need not, simply be a copy of a conventional, commodity OS), virtual system memory 312 (VMEM), a virtual disk 314 (VDISK), optional virtual peripheral devices 340 (VDEVICES) and drivers 322 (DRVS) for handling the virtual devices 340, ***all of which are implemented in software to emulate the corresponding components of an actual computer***. Although the key components of only one VM are illustrated in FIG. 1, the structure of other VMs will be essentially identical.

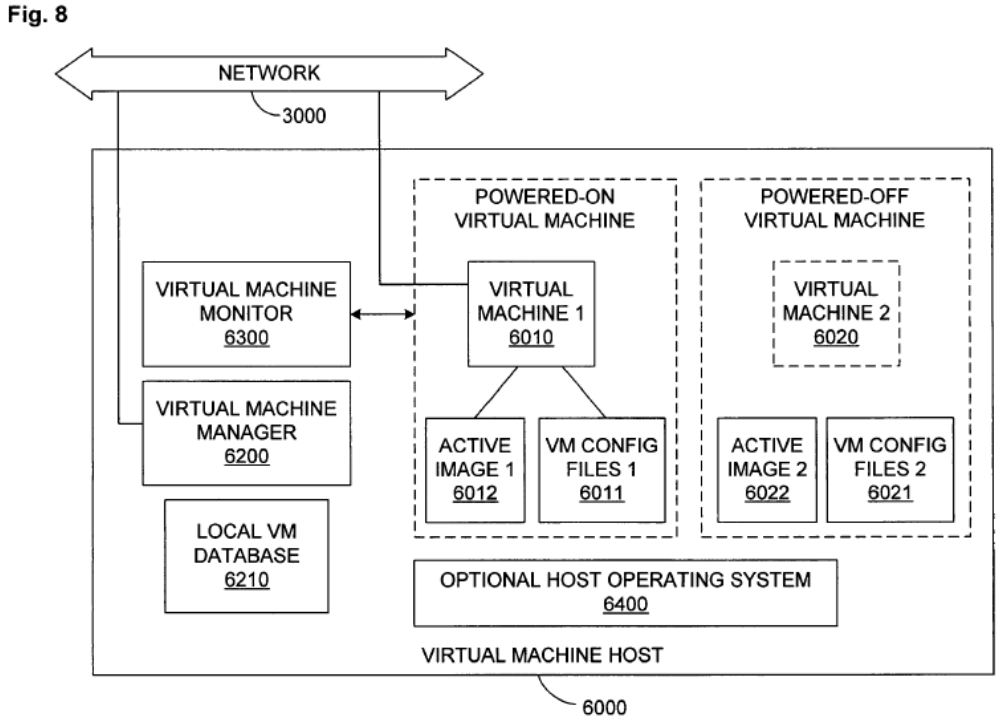
Id., 6:44-61 ("If the VM is properly designed, then the applications (that is, the user of the applications) will not 'know' that they are not running directly on 'real' hardware."). Given Khandekar's teachings, a POSITA would have understood that each VM emulates a computer platform because the applications running on the virtual machine believe they are running directly on real hardware. This was the conventional and known functionality of virtual machines.

116. Thus, for the reasons explained above, Khandekar renders this limitation obvious. *See* §VI.A.1.

Ground 2: Le's teachings, when combined with Khandekar, further render this limitation obvious

117. To the extent Patent Owner contends that this claim limitation [1d] requires that *each* image instance of a VMM provide a plurality of virtual machines, Khandekar teaches this as explained above. The combination of Khandekar and Le further renders this obvious. *See* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le).

118. Le provides additional motivation to apply Khandekar's teaching of using "a single VMM to act as the interface to all VMs." *See, e.g.*, Ex. 1004, 7:28-31. For example, Le teaches a single virtual machine monitor and a single virtual machine manager that support a plurality of virtual machines on a host. *See, e.g.*, Ex. 1005, FIG. 8 (6200 and 6300):



"[T]wo virtual machines 6010 and 6020 are shown" but "[a]ny number of virtual machines may be loaded onto the host 6000, limited only by the capacity of the host." Ex. 1005, 65:43-50. Therefore, the combination of Khandekar and Le teaches that when executed on the applications nodes, the image instances of the virtual machine manager provide a plurality of virtual machines. By 2006, VMM systems that were known in the art supported multiple VMs per instance, including commercially available software such as VMware GSX and ESX Server (*see* Ex. 1005, 65:65-66:10) and the Xen hypervisor, an x86 virtual machine monitor, (*see* Ex. 1011 at 1, 5, FIG. 1). *See also* Ex. 1016 ¶[0076] ("VM manager 20 creates a plurality of virtual machine environments 40"). Therefore, Khandekar and Le teach a known and conventional usage of VMM and VM software, and a POSITA would have found it obvious and been motivated to combine those teachings.

119. Le teaches that the virtual machine monitor and virtual machine manager are executed on the host computers, where the image instances of the each virtual machine monitor and virtual machine manager provide a plurality of virtual machines, with each of the plurality of virtual machine operable to provide an environment that emulates a computer platform. *See supra* §VI.A.3[1c]; Ex. 1005, 65:65-66:10 ("When a virtual machine is powered on, a virtual machine monitor program 6300 controls it and manages the interactions between the

software—commonly called 'guest software'—running inside of the virtual machine and the host's physical resources, such as hardware devices."), 66:11-32:

> The virtual machine manager 6200 is typically also responsible for creating, destroying and maintaining virtual machine files residing on the host 6000. It may use a local registration database file 6210 to keep track of the virtual machines hosted on the computer. In order to integrate with the UCMS, the virtual machine monitor 6300 must expose a network interface to the UCMS imaging server 2101 and this interface needs to expose a minimum set of required service functions.
>
> The first function returns the set virtual machines registered on the host 6000 and the properties of each virtual machine, such as its current power state and the contents of its configuration file. …

120. A POSITA would have found it obvious to combine Le's teachings with Khandekar's teachings because both are VMware patents directed to features of virtual machine systems, and Le teaches particular VMM software provided by the same company, as explained above. *See supra* §VI.A.2.

121. Therefore, the combination of Khandekar with Le's additional teachings further renders limitation [1d] obvious. *See* §VI.A.2.

[1e]	**(ii) a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines; and**

122. Khandekar teaches or suggests this limitation. For example, Khandekar teaches or suggests a software image repository comprising non-

transitory, computer-readable media operable (*see supra* §VI.A.3[1b]) to store a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines. *See e.g.*, Ex. 1004, 18:56-19:2 ("The applications library 850 is a repository of the software and data needed to install an application 856 in a VM."), 10:15-26 ("… for pre-built VMs, a list of available staged VMs, along with a description of their respective (virtual) hardware platform, system software, and pre-installed applications (if any) …"), 14:4-18 ("A brief description of the operating system, service pack and set of applications installed in each instantly deployable VM is preferably also included.").

123. This limitation may be satisfied by a plurality of virtual disk image files (*see supra* §V.A (explaining claim construction for this term)), which Khandekar teaches three ways: with pre-built, custom-built, and instantly-deployable VMs. Each teaching independently renders limitation [1e] obvious by teaching a plurality of virtual disk image files. For pre-built and instantly-deployable VMs, Khandekar teaches that each virtual disk image file includes an operating system and application data, as explained below.

124. To the extent Patent Owner contends the claimed software applications are somehow distinct from the virtual disk image file, Khandekar would still render this limitation obvious through its teaching of custom-built VMs.

For custom-built VMs, Khandekar teaches that each virtual disk image file includes an operating system, and application images, in the form of application installation files, are stored in an application library and copied to hosts, as explained below.

125. Khandekar presents the user with three choices for storing and deploying image instances. *See* Ex. 1004, 9:10-25 (emphasis added):

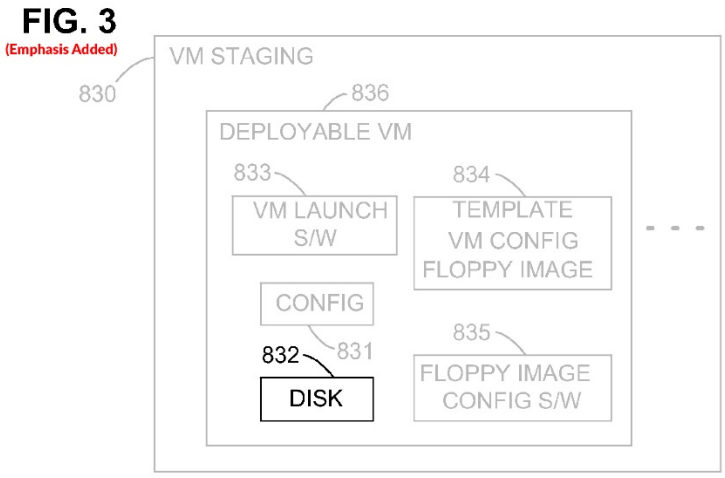
> … The invention provides the user with three choices:
>
> **1)** Clone an existing, **pre-built model VM** and configure a few parameters, such as host name, IP address, etc.
>
> **2)** Install an operating system and/or one or more applications to a VM created from scratch and then configure them. Such a VM is referred to below as a **"custom-built" VM**.
>
> **3) Instantly provision** a VM by resuming it from a suspended state. Such a VM is referred to below as a an "instant-deploy" VM.

See also, *infra* §VI.A.3[1g]. A POSITA would have been motivated to apply all three approaches because Khandekar teaches using them together. *See supra* §VI.A.1.

126. **(1) Pre-Built, Staged VMs.** Khandekar teaches that its provisioning server includes a plurality of images of pre-built VMs that are stored in a repository (e.g., library 830). *See, e.g.*, Ex. 1004, 10:27-29 ("a library 830 of staged VMs"), 16:43-63, 18:16-23 ("Staging VMs: This data structure is required for

provisioning pre-built or custom-built VMs and contains … [l]ocation of the VM in the VM staging repository 830."), 22:17-22, FIGS. 2-3. The pre-built VMs include operating systems and application data installed on their virtual disk image files. *See, e.g.*, Ex. 1004, 15:40-16:12 ("… pre-built VMs will be stored with their ***operating systems and applications installed*** … In the first step, a model VM is created and the necessary OS and applications are installed in it.") (emphasis added), 5:25-36 ("…the invention makes this possible by allowing the user either to select one of a set of 'model' virtual machines that are ***pre-built*** with such components as virtual hardware, ***an operating system and one or more applications***, or to specify desired components that are then assembled into a virtual machine….") (emphasis added), 14:19-25, 18:16-23, 10:15-26, 8:50-55 ("A 'model virtual machine' is a virtual machine (VM) that contains a certain configuration of guest operating system 320, service pack, and applications 360, where the applications have been tested and are known to work correctly."), 23:54-24:27, FIG. 3.

127. Pre-built VMs also include a virtual disk image file, for example as indicated by 832 "disk" in Figure 3. *See* Ex. 1004, FIG. 3:



See also Ex. 1004, 15:62-16:12, 16:21-29. The staged VMs stored in the library each include virtual disk image files along with configuration files and supporting configuration and launch components. *See* Ex. 1004, 16:47-63 (emphasis added):

> For each VM, the data sets kept in the VM staging server 830 are illustrated in FIG. 3 and are as follows:
>
> 1) The VM configuration ("config") file 831.
>
> **2) VM disk files 832.**
>
> 3) A launch software module 833, which is installed inside the VM and resides on the disk 832.
>
> 4) A template VM configuration floppy image 834, which is used by a provisioning engine 810 at the time of deployment to provide the data to the VM launch software 833 at the time of boot to enable the VM to configure itself according to externally specified parameters. This is discussed in more detail below.

> 5) A software module 835 for creating a floppy image starting from the template floppy image 834 and customized according to the user input. This is also discussed in more detail below.

The configuration file, template VM configuration floppy image, and software modules allow parameters such as "computer hostname, IP address, etc." to be configured on a host computer without modifying the VM disk files. *See, e.g.*, *id.*, 19:19-20:8, 16:47-63. A POSITA would have found it obvious to follow these teachings because they simplify the deployment process and allow a VM disk file, which is often large, to be reused without modification. *See, e.g.*, *id.*, 19:19-20:8, 16:47-17:13. Khandekar teaches a hierarchal disk structure where VM disk files in the inner-layer are immutable and shared. *See id.*, 16:64-17:12.

128. A "VM Staging" process is used to create new VM images, including for pre-built, custom-built, and instantly-deployable VMs, which are then stored in a library and database for future deployment. *See, e.g.*, Ex. 1004, 15:29-40 ("When the user has finished making the above selections, he submits the request to the provisioning engine 810, which creates a new VM by the process described below. … VM Staging 830—FIG. 3"), 16:43-49 ("After the above steps are completed, the guest operating system in the VM is shut down and the VM is powered down (for the VM, a software procedure). This VM is then ready to be deployed and is stored as one of the staged VMs 830. This VM may then also be added to the database 890.").

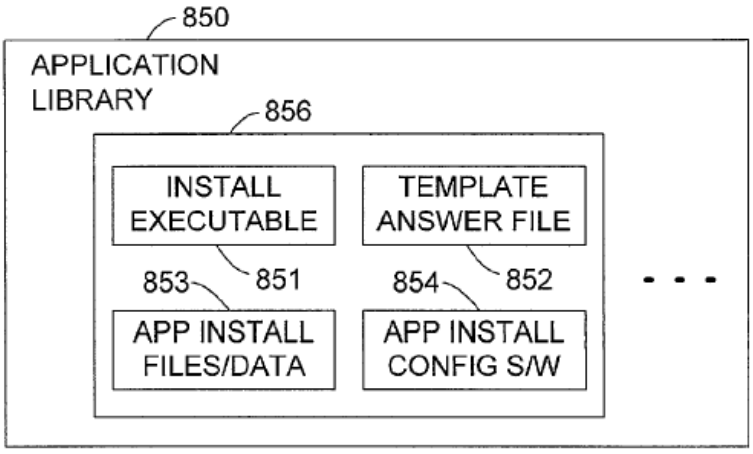
129. Staged VMs are prepared in a "two-step process, namely, creation of the model VM and then staging of this VM." Ex. 1004, 15:40-16:12. "In the first step, a model VM is created and ***the necessary OS and applications are installed*** in it." *Id.* (emphasis added). "After the above steps are completed, the guest operating system in the VM is shut down and the VM is powered down (for the VM, a software procedure). This VM is then ready to be deployed and is stored as one of the staged VMs 830." *Id.*, 16:43-52. "This VM may then also be added to the database 890." *Id.*

130. Khandekar teaches that "any number of deployable VMs may be staged." Ex. 1004, 15:46-50. A POSITA would have been motivated and found it obvious to follow this teaching by staging a plurality of VM images in the "VM staging repository 830" because doing so would have increased the speed and efficiency in which VMs can be deployed—particularly for frequently-deployed VMs or a set of VMs related to frequently-requested configurations. Indeed, this is a preferred embodiment of Khandekar. *See* Ex. 1004, 4:36-47 ("In a preferred embodiment of the invention, ***a plurality of pre-configured VMs having different configurations are pre-stored***. A requester (which will usually be a human user but may also be a computer program) then selects one of the pre-configured VMs, which is then automatically prepared for deployment.") (emphasis added), FIG. 7. It would have been obvious to keep staged VMs for a period of time after they

have been deployed, so they are readily accessible if they are requested again. In this manner, copies of frequently requested VMs are readily accessible to be deployed. This was a common approach at the time.

131. **(2) Custom-Built VMs.** Custom-built VMs may have additional applications installed. Khandekar teaches images of software applications stored in an applications library 850 that are executable on VMs. *See e.g.*, Ex. 1004, 18:56-19:2 ("The applications library 850 is a repository of the software and data needed to install an application 856 in a VM. Since the applications are installed by the provisioning engine 810 during deployment phase (see below), they are installed unattended, that is, without any human intervention."), 10:27-42 ("The heuristics engine 880 also selects the applications from a library 850 of available applications to install in the newly created VM."), 11:50-58 ("… library 850 … contain[s] code for any collection of the myriad of existing applications, from web servers, database servers, spreadsheets and word-processing programs, to games and browsers."), 12:13-23 ("… to store in the library 850 different copies of applications, with different settings for the different possible operating systems, different copies of operating systems configured for different processors, etc."), FIG. 5:

FIG. 5



132. Custom-built VMs are staged using the process described above for pre-built VMs. *See* Ex. 1004, 15:29-17:29. Like pre-built VMs, custom-built VMs include VM disk files that are stored in the provisioning server and copied to hosts when deployed. *See id.*, 16:43-17:13. For custom-built VMs, the virtual disk image only includes the operating system. *See* Ex. 1004, 15:41-46 ("…custom-built VMs will be stored with just their operating systems installed."). Application images, in the form of application installation files, are stored in the provisioning server's application library and copied to hosts. *See id.*, 19:20-20:30 (emphasis added):

> … **The temporary usage area 866 is also used to install additional applications in a custom-built VM.** As explained above, in the discussion of the applications library, **application install software is copied** to the temporary usage area (shown as 862) and the custom answer file for the application is also written into the temporary usage area (864). The VM launch software 833 then establishes a connection (a shared folder, for example) with the temporary usage area on the host server and executes the application install software 862. This

> software takes the answer file data 864 and installs the applications in the VM. In this embodiment of the invention, **the temporary usage area 860 will reside on the host server where the VM is being deployed**.

See id., 23:28-51, 21:25-64, 4:42-51. Applications are installed without human intervention. *See* Ex. 1004, 18:56-19:2 (emphasis added):

> Since the applications are installed by the provisioning engine 810 during deployment phase (see below), **they are installed unattended, that is, without any human intervention**. Application vendors typically provide ways to install their applications unattended. The common procedure for unattended installation involves running an executable installation file 851 that takes input from a text file, called an "answer file," that specifies the parameters for installation for that application (for example, the folder to install it in) and the installation files/data 853 that need to be copied to the machine (here, virtual machine) that the application is to be installed on.

133. Khandekar teaches that "any number of deployable VMs may be staged." Ex. 1004, 15:46-50. A POSITA would have been motivated and found it obvious to follow this teaching by staging a plurality of VM images in the "VM staging repository 830" because doing so would have increased the speed and efficiency in which VMs can be deployed—particularly for frequently deployed VMs or a set of VMs related to frequently requested configurations. For example, it would have been obvious to keep staged VMs for a period of time after they

have been deployed, so they are readily accessible if they are requested again. In this manner, copies of frequently requested VMs are readily accessible to be deployed. This was a common approach at the time, and it applies to custom-built VMs because it avoids the need to re-build custom VMs that are frequently requested.

134. **(3) Instantly-Deployable VMs.** Instantly-deployable VMs can be deployed and then resumed in a fully-booted state. *See, e.g.*, Ex. 1004, 17:30-50. "[T]hese ready-to-deploy VMs (one of which is indicated with reference number 846) are created as follows. First, a new VM is created according to the steps outlined above in the section on VM Staging." Ex. 1004, 17:30-50 (referring to the VM staging steps discussed above for pre-built VMs). "In the first step, a model VM is created ***and the necessary OS and applications are installed in it***." *Id.*, 15:49-52 (emphasis added). As discussed above, the staging process creates a VM with "VM disk files 832" and various configuration and software files/modules. *See* Ex. 1004, 16:43-63. Therefore, as with pre-built VMs, the instantly-deployable VMs also include a virtual disk image file with an operating system and application data.

135. Unlike the other approaches, instantly-deployable VMs are suspended in a fully-booted state. *See* Ex. 1004, 17:30-50 ("The VM's guest OS 320 is then shut down and the VM's virtual disk 314 is switched to a non-persistent mode. The

VM is then powered on again (once again, a software procedure) and, when it is fully booted, it is suspended."). Khandekar teaches that "[a] virtual disk file 841 is preferably stored in a network share folder so that it is accessible from any machine in the LAN." *Id.* Since the provisioning server is a central repository on the network, a POSITA seeking to apply this optional teaching would have found it obvious, based on Khandekar's teachings, to store the virtual disk on a network share folder on the provisioning server. *Id.*, 18:1-5. "A REDO log file 843, as well as a suspend-to-disk image file 844 and a VM config file 842 are then preferably also stored as part of the instant-deploy VM 846." *Id.*, 17:30-50. "The VM will then be ready to be deployed (see the deployment section below), and may also be added to the database 890." *Id.*

136. Khandekar teaches that images of instantly-deployable VMs are stored, with an operating system and applications, in the "instantly-deployable VMs repository 840." Ex. 1004, 18:1-15 ("Instantly deployable VMs: This is a data structure indexed by the name of the VM configuration, and contains the following fields: … b) Operating system … d) List of applications available … f) Location of the VM in the instantly-deployable VMs repository 840."), 14:4-18 ("A brief description of the operating system, service pack and set of applications installed in each instantly deployable VM is preferably also included."), 18:6-15, FIG. 4. A POSITA would have found it obvious to store more than one instantly-

deployable VM because Khandekar teaches a repository, which were typically used to store multiple items. Furthermore, Khandekar teaches tracking the OS, service pack, and applications on each instantly-deployable VM, implying that a plurality of such VMs are stored with different OS and applications. Moreover, it would have been obvious to store frequently-deployed configurations, as explained above for pre-built and custom VMs. Therefore, Khandekar teaches or suggests that the provisioning server stores a plurality of instantly-deployable VMs.

137. **Other Limitations in [1e]**. Khandekar teaches the image instances of a plurality of software applications that are executable on the plurality of virtual machines. As explained above, each pre-built, custom-built, and instantly-deployable VM includes a virtual disk image file with an operating system and applications that are executable on the virtual machines. *See also, e.g.*, Ex. 1004, 6:43-56 ("Of course, most computers are intended to run various applications, and VMs are usually no exception. Consequently, by way of example, FIG. 1 illustrates one or more applications 360 installed to run at user level on the guest OS 320; any number of applications, including none at all, may be loaded for running on the guest OS, limited only by the requirements of the VM itself. Usually, no modifications of any kind are needed in order to install applications on a VM as opposed to on a 'real' computer. If the VM is properly designed, then the applications (that is, the user of the applications) will not 'know' that they are not

running directly on 'real' hardware. Of course, all of the applications and the components of the VM are instructions and data stored in memory, just as any other software.").

138. Moreover, a POSITA would have found it obvious to store a plurality of image instances of a plurality of software applications because "many applications have settings and parameters that depend on the operating system they will be running on, and an operating system will have parameters that depend on which processor it is installed on." *See* Ex. 1004, 12:13-23 ("One way to overcome these complications is to store in the library 850 different copies of applications, with different settings for the different possible operating systems, different copies of operating systems configured for different processors, etc.").

139. Thus, for the reasons explained above, Khandekar's teachings render this limitation obvious. *See* §VI.A.1.

[1f]	**a control node that comprises an automation infrastructure to provide autonomic deployment of the plurality of image instances of the virtual machine manager on the application nodes by causing the plurality of image instances of the virtual machine manager to be copied from the software image repository to the application nodes and**

140. Khandekar teaches or suggests this limitation. For example, Khandekar teaches or suggests a control node (e.g., "provisioning server") that comprises an automation infrastructure (e.g., "provisioning engine,"

"heuristics/rules engine," and supporting functionality in the provisioning server) to provide autonomic[3] deployment of the plurality of image instances of the virtual machine manager (e.g., VMM) on the application nodes (e.g., "hosts") by causing

[3] The '138 patent uses both "autonomic" and "automatic" to refer to automated processes that do not require human intervention. For example, compare Ex. 1001, 8:13-23 (emphasis added):

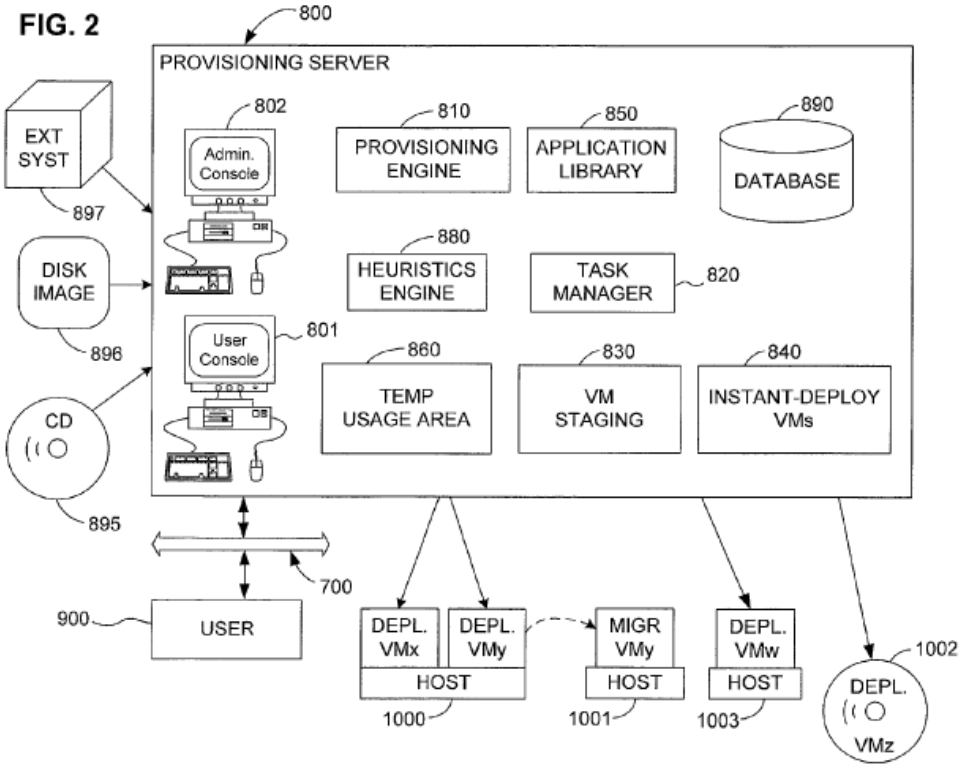
> As will be described in detail below, control node 12 may ***automatically*** add unallocated nodes from free pool 13 to a tier when more processing capacity is needed within the tier, remove nodes from a tier to the free pool when the tier has excess capacity, transfer nodes from tier to tier to meet processing demands, or replace failed nodes with nodes from the free pool. Thus, computing resources, i.e., computing nodes, may be ***automatically*** shared between tiers and domains within the fabric based on user-defined policies to dynamically address high-processing demands, failures and other events.

with *id.*, 35:58-67 (emphasis added):

> After administrator 20 activates the tier, based on the current state and goals of distributed computing system 10, control node 12 ***autonomically*** selects a virtual machine from free pool 13, associates the virtual machine with a slot in the tier, identifies the virtual disk image file (image instance) associated with the particular slot, copies the virtual disk image file to the local hard disk of the physical node on which the virtual node resides or the an associated SAN, and finally boots the virtual machine from the virtual disk image file.

See also id., 7:5-16, 20:4-6.

the plurality of image instances of the virtual machine manager to be copied from the software image repository to the application nodes. *See, e.g.*, Ex. 1004, 5:25-36 ("The invention enables users to configure, deploy and get access to substantially 'customized' computer systems … with no need for human intervention."), 9:9-25, 23:54-24:27, FIG 2:



141. Khandekar's provisioning server is a node in the network that controls the automated deployment of VMM and VM images from the provisioning server's software image repository (*see supra* §VI.A.3[1b]-[1e]) to hosts. *See, e.g.*, Ex. 1004, 5:25-36, 9:9-25, 23:28-51, FIGS. 2, 7. For example, Khandekar teaches "[a] requester (which will usually be a human user but may also be a computer

program) then selects one of the pre-configured VMs, which is then ***automatically prepared for deployment***." *Id.*, 4:36-41 (emphasis added). "In most embodiments of the invention, there is a plurality of physical hosts on which VMs can be deployed. In this case, according to another aspect of the preferred embodiment of the invention, ***the status of the hosts is monitored and the host on which a configured VM is to be deployed is selected heuristically***. A deployed VM may also be migrated from one host to another based on the monitored status of the plurality of hosts." *Id.*, 4:58-65 (emphasis added). Le also teaches monitoring status and performance information. *See* Ex. 1005, 68:21-24 ("… the agent 7300 can also report status and system performance information to the imaging server 2101, allowing the UCMS to detect the presence of the agent and to monitor the deployed computer's health.").

142. Khandekar teaches that the deployment of images is automatic. For example, after a request is made, a heuristics engine "applies heuristics/rules to select the most appropriate VM from a library 830 of staged VMs." Ex. 1004, 10:27-29. "At least one host computer 1000, 1001 is made available to the provisioning server, and the heuristics engine 880 also selects which host machine 1000, 1001 is most appropriate to provision and deploy the VM on." *Id*, 10:29-38. "Known expert systems, 'intelligent assistants,' and other knowledge-based systems and engines that can be embedded in other products and loaded with

domain knowledge may be included in the heuristics engine 880 to implement the various decision functions." *Id,* 10:38-42. Khandekar teaches the automation infrastructure provides automatic deployment of the VM images to the host. *See infra* §VI.A.3[1g].

143. Khandekar further teaches sending a VMM with a VM to be installed on a host. *See* Ex. 1004, 8:36-44 ("… when a VM is installed on a host, then a corresponding VMM is either already installed on the host, ***or is included and installed together with the VM***.") (emphasis added), 13:25-28. A POSITA would have found it obvious and been motivated for the provisioning server to provide a VMM together with the VM because, as explained previously, it avoids the need for manual installation and provides flexibility and is advantageous for distributed, cloud computing systems. *See supra* §VI.A.1, §VI.A.2, §VI.A.3[1c] (explaining how and why a POSITA would have been motivated to apply Khandekar's teaching of including a VMM together with the VM). When the provisioning server is able to provide both a VMM and a VM to a host, it allows the server to automatically and dynamically initialize a host, identified by the heuristics/rules engine, with a VMM so that it is ready to run VMs without the need for a human sysadmin, improving scalability in large cloud systems. *See id.*

144. "As is mentioned above, a VMM is typically included for each VM in order to act as its software interface with the underlying host system." *Id.*, 13:17-

35. “Several products are available from VMware, Inc. of Palo Alto, Calif., that configure conventional computers, both stand-alone desktop computers and multi-user servers, to load and run multiple VMs.” *Id.* A POSITA would have found it obvious and been motivated to autonomically deploy image instances of VMMs from the provisioning server to the hosts because Khandekar teaches (1) automatic deployment of VM images, and (2) providing a VMM together with the VM. *See, e.g.*, Ex. 1004, 4:36-41, 4:58-65, 10:27-42, 8:36-44, 13:17-35. Therefore, it would have been obvious based on Khandekar’s teachings that the provisioning server is a control node within the network that includes an automation infrastructure to provide autonomic deployment of the plurality of image instances of the virtual machine manager on the application nodes.

145. Khandekar teaches that the autonomic deployment of images is accomplished by copying the image from the provisioning server’s repository to the host computer. As explained below, Khandekar teaches copying VMs from the provisioning server to host computers. *See infra* §VI.A.3[1g]. Khandekar further teaches that a VMM “is included and installed together with the VM.” *See* Ex. 1004, 8:36-44, 13:25-28. Since the VMM is included with the VM, it is copied along with the VM to the host computer. Moreover, it would have been obvious to first copy the VMM image to the host computer so that it can be installed on the host computer, as taught by Khandekar. This conventional usage of VMM images

was known and used in the art, and thus a POSITA would have found it obvious and had a reasonable expectation of success when applying Khandekar's teaching in this manner. *See, e.g.*, Ex. 1014 ¶[0062] ("The hypervisor also may be loaded first or part of an OS-virtual machine manager image. It may be essentially part of a firmware image."); Ex. 1016 ¶¶[0049], [0043]. Khandekar teaches autonomic deployment of images because, as explained above, it teaches an automated deployment that does not involve human intervention. *See e.g.*, Ex. 1004, 18:56-19:2 ("Since the applications are installed by the provisioning engine 810 during deployment phase (see below), they are installed unattended, that is, without any human intervention."), 5:25-36 ("The invention enables users to configure, deploy and get access to substantially 'customized' computer systems … with no need for human intervention."), 4:36-41 (VMs are "automatically prepared for deployment.").

146. Claim 1 recites "autonomic deployment" but does not recite autonomic *requests* for VMs. In any event, Khandekar teaches both. For example, Khandekar teaches that a user may be "any person or entity who wants one or more VMs to be created and deployed according to their specifications," but "could also be a software application or tool." Ex. 1004, 9:49-52. "Software applications acting as a user interact with the provisioning server using a well-defined programmatic interface (API), which may be designed in a known manner." *Id.*, 9:56-59, 22:53-

67 ("…as long as a protocol is agreed upon (in terms of a well-defined programmatic interface or API) by the provisioning server and an external (possibly remote) computer system, then the external computer could itself input VM provisioning requests to the provisioning server. …"). Thus, Khandekar teaches autonomic requests for VMs by software applications that act as users. *See id*. A POSITA would have found it obvious and been motivated to apply these teachings because it increases automation, which was one of the goals behind Khandekar's provisioning server, and Khandekar teaches that the use of software agents and related APIs for automatically requesting VMs was known in the art. *See id.*, 9:49-64, 22:53-67. Khandekar provides express motivation to apply automated teachings, explaining benefits for testing environments. *See, e.g.*, *id.*, 22:53-67 (emphasis added):

> This arrangement would be beneficial in the context of testing, in which a very large number VMs having different hardware/OS/application combinations need to be provisioned. The external computer could then be programmed to request all the needed combinations, run the VMs, and log the results. Using the invention, such testing of many configuration combinations could therefore be made ***substantially wholly automatic***.

147. Thus, for the reasons explained above, Khandekar renders this limitation obvious.

Ground 2: Le's teachings, when combined with Khandekar, further render this limitation obvious

148. Khandekar teaches autonomic deployment of VM and VMM images, as explained above for Ground 1. Le provides additional desk imaging teachings that complement and elaborate on Khandekar's teachings with techniques for copying images from a repository to remote computers, further rendering this limitation obvious. *See* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le).

149. For example, Le teaches techniques for copying images from a central repository to destination computers. As explained previously, a POSITA would have found it obvious to use Le's disk image teachings to implement Khandekar's teaching of providing the VMM from the provisioning server to the host. *See supra* §VI.A.3[1c]. Le teaches techniques where "[d]isk imaging can also be used to clone a computer; an image of a first computer can thus be deployed to other computers." Ex. 1005, 2:19-21. Disk images include the operating system and software applications, which would include Type 1 and 2 VMMs, respectively. *See, e.g.*, Ex. 1005, 8:10-23, 10:55-62, 22:32-44; *supra* §VI.A.3[1c]. Therefore, as explained previously, a POSITA would have found it obvious to use Le's disk imaging techniques to install a VMM—including the operating system and software applications such as Type 1 and Type 2 VMMs, respectively—onto remote computers over a network. *See supra* §VI.A.3[1c]. This includes using

imaging for VMware software, such as ESX and GSX software, which like all other software installed on a computer would have been included in the disk imaging process. To the extent Patent Owner attempts to distinguish between virtual machine monitor and managers, both were components of VMware software; therefore, it would have been obvious to include the virtual machine monitor and managers taught by Le in the images described above.

150. Le teaches that disk imaging was conventionally used in the art to clone, i.e., copy, an image of a computer, including OS and applications, onto other machines. *See, e.g.*, Ex. 1005, 2:19-21 ("Disk imaging can also be used to clone a computer; an image of a first computer can thus be deployed to other computers."), 16:29-30 ("Disk imaging is often used as a cloning technology for rapid computer deployment."), 10:55-62 (emphasis added):

> **An image is often used to make multiple clones of a base computer.** The approach is to capture an image from the base computer's disk, and then deploy the same image to multiple destination computers. Before the initial image is captured, the base computer is **configured with an operating system and common set of software applications that are required on all clones**. Any computer deployed from this image would inherit the same set of software.

See also id., 54:14-21 ("A computer management system can use disk imaging to quickly provision a bare-metal computer on the network with a complete software

stack consisting of an operating system and a set of core applications. … A single template image, or a small set of core template images, is used to create clones.").

151. Le describes in detail the process of creating an image from a source machine, and explains that deployment of an image is the inverse process whereby an image is copied to the destination computer. *See, e.g.*, Ex. 1005, FIGS. 3-5:

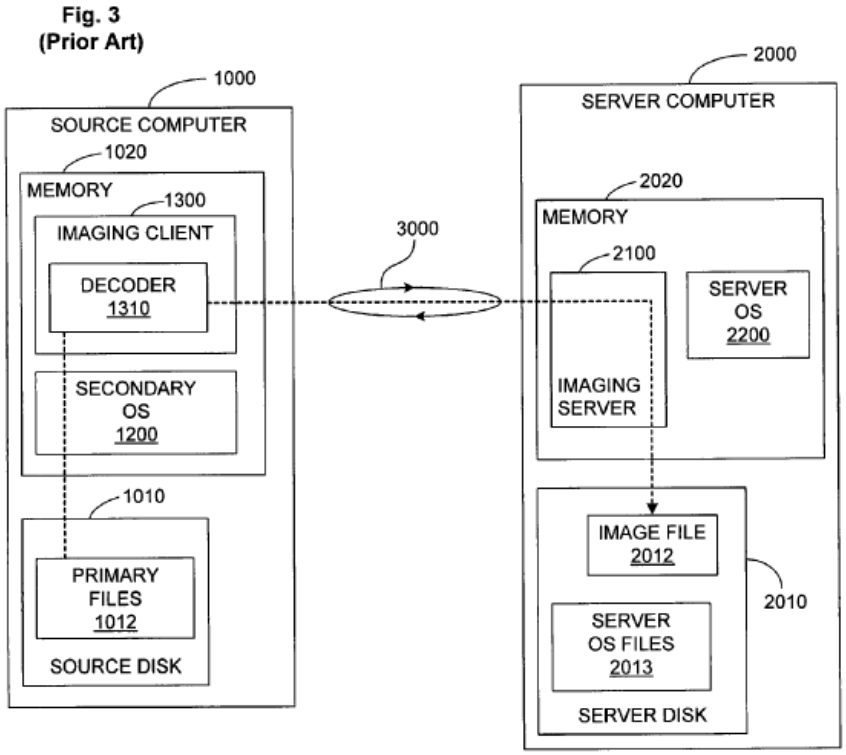


Fig. 4

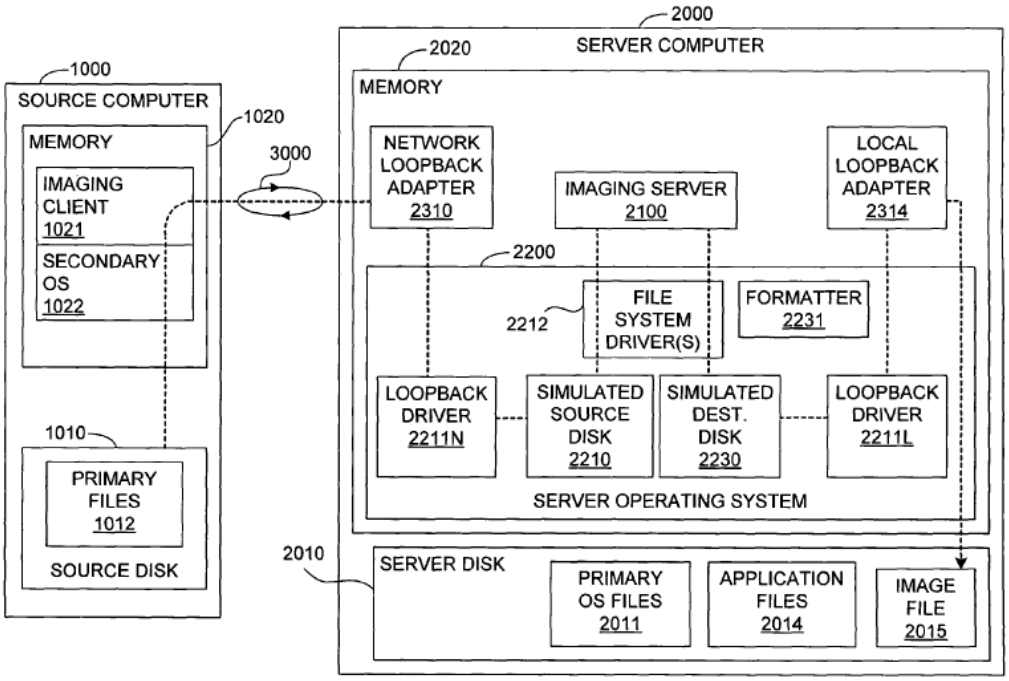
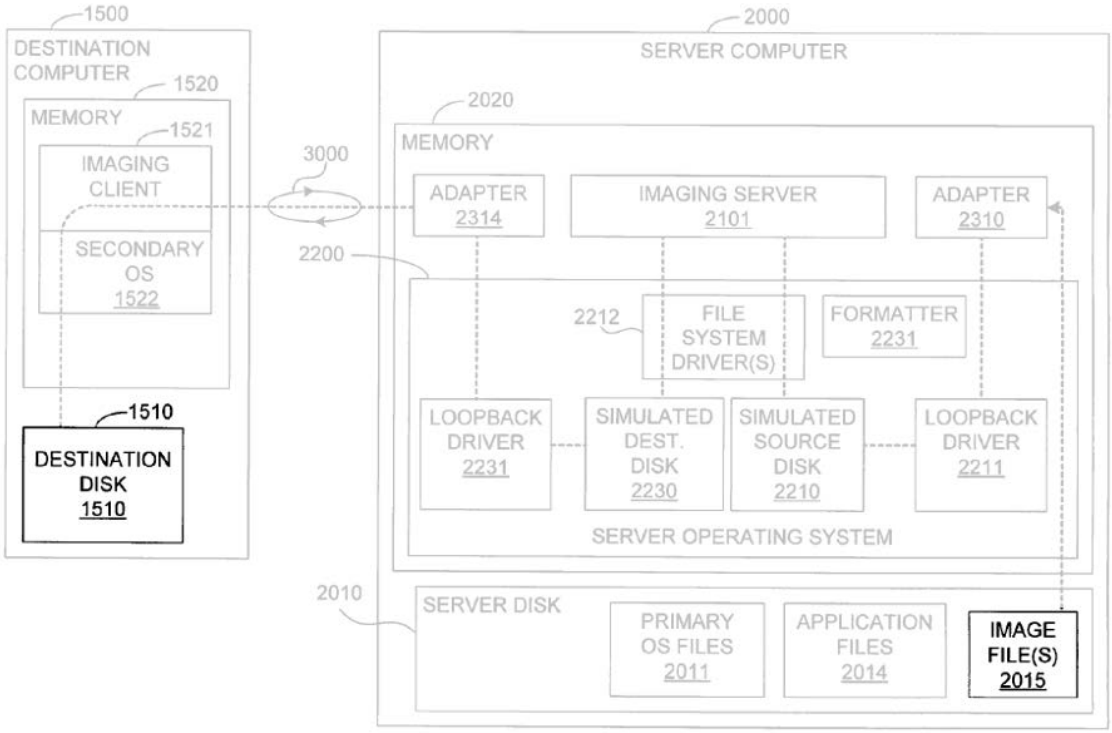


Fig. 5 (Emphasis Added)



See Ex. 1005, 26:52-27:9 ("… The invention's image deployment process is illustrated in FIG. 5 and is symmetrical to the novel image capture process described above. … The adapters 2310, 2314 and the file system drivers 2212 will perform the same functions as for image capture, but in the reverse direction. … ***Finally, the imaging server copies files and directories from the source file system to the destination file system***, dismounts the two simulated disks 2210, 2230, and reboots the destination computer 1500 to allow it to load the new operating system deployed from the image 2015.") (emphasis added); *see also id.*, 72:14-22 ("… It then copies all files and folders from the source to the destination."), 20:25-37 ("A virtual disk can also be used as a regular disk image for physical computers, i.e., it can be archived, cataloged, then later deployed to one or multiple physical computers for backup/restore or cloning purposes."), FIG. 9; *infra* §VI.A.3[1g].

152. Le teaches an "exemplifying embodiment," the "Universal Computer Management System (UCMS)," (Ex. 1005, 59:21-27), which is similar to Khandekar's provisioning server. *See supra* §VI.A.2. The UCMS manages and deploys images to virtual machines and physical computers. *See, e.g.*, Ex. 1005, 65:20-33, 59:29-36. To guide deployment to physical computers, the UCMS includes functionality that analyzes the hardware configuration of new computers when they are registered. *See* Ex. 1005, 70:12-35:

> Deploying to Physical Computers
>
> In order to become registered with the UCMS, a new physical computer must first boot into the UCMS secondary software stack 4100, allowing the UCMS server 2101 to detect the computer and to add a record for it in the registration database 4004 (see Physical Computer Control). …
>
> The registration process causes the imaging client 1021 running from the secondary stack 4100 to analyze the computer's hardware configuration and transmit it over the network 3000 to the UCMS server 2101. …

153. A POSITA would have found it obvious to use these teachings from Le to address Khandekar's teaching that a VMM may be provided to a host "as long as the characteristics of the host on which the VM is to be installed are known." *See* Ex. 1004, 13:17-35. Thus, it would have been obvious, when combining Khandekar and Le, for the provisioning server to deploy VMM images to physical computers and VM images for virtual machines, using the hardware configuration of the host computer to guide the automation process and send an appropriate image to the host computer. *See, e.g.*, Ex. 1005, 65:20-33, 59:29-36, 70:12-35.

154. Therefore, the combination of Khandekar with Le's additional teachings further renders limitation [1f] obvious.

[1g]	**to provide autonomic deployment of the plurality of image instances of the software applications on the virtual machines by causing the plurality of image instances of the software applications to be copied from the software image repository to the application nodes.**

155. Khandekar teaches or suggests this limitation. For example, Khandekar teaches or suggests a control node (e.g., "provisioning server") that comprises an automation infrastructure (e.g., "provisioning engine," "heuristics/rules engine," and supporting functionality in the provisioning server) to provide autonomic deployment of the plurality of image instances of the software applications on the virtual machines by causing the plurality of image instances of the software applications to be copied from the software image repository to the application nodes (e.g., hosts). *See supra* §§VI.A.3[1e]-[1f]; Ex. 1004, 23:54-24:27, FIGS. 2, 7:

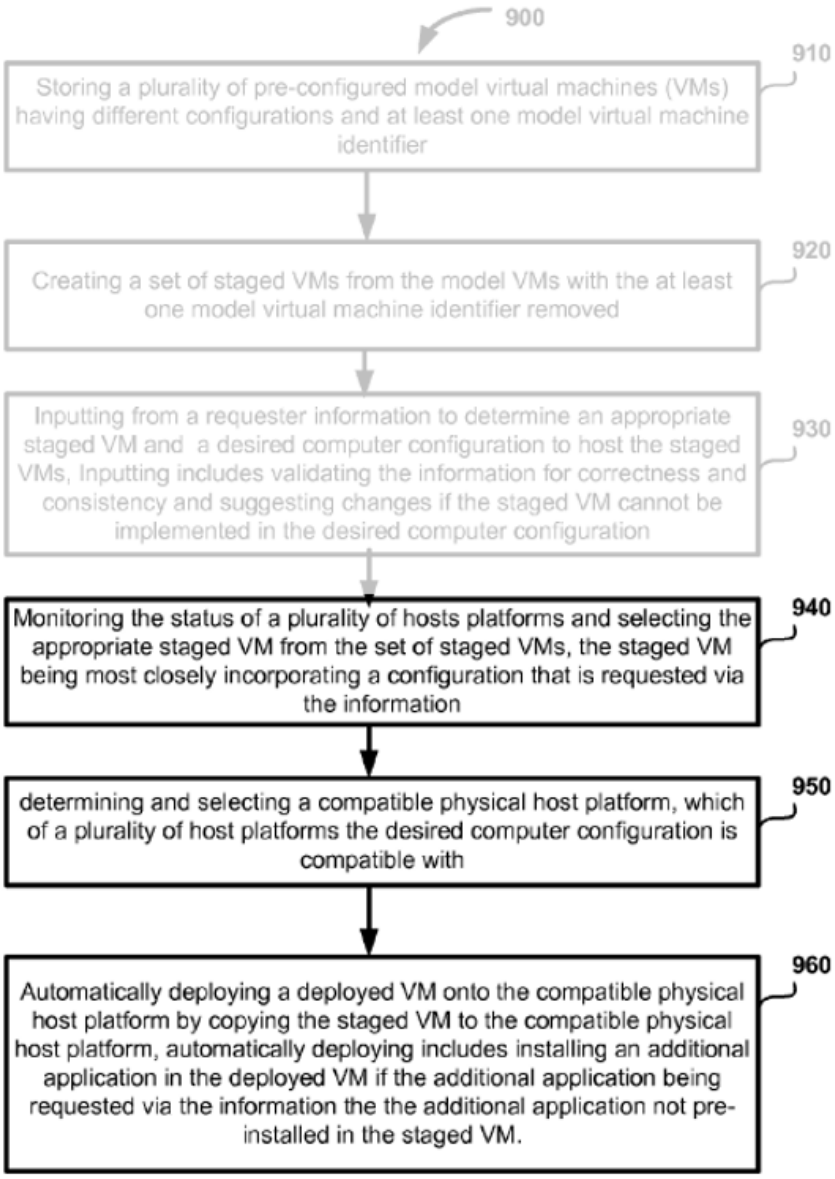


FIG. 7
(Emphasis Added)

156. Khandekar's provisioning server is a node in the network that controls the automated deployment of VM images from the provisioning server's repository to host computers. *See supra* §VI.A.3[1f]; Ex. 1004, 4:36-41, 4:58-65, 10:27-42. The autonomic deployment process was described above. *See supra* §VI.A.3[1f].

For example, with respect to Fig. 7, "a VM is automatically deployed onto the compatible physical host platform by copying the staged VM to the compatible physical host platform." *See id.*, 23:28-51:

> At step 940, the status of a plurality of hosts platforms is monitored and the appropriate staged VM is determined and selected from the set of staged VMs. The staged VM most closely incorporates a configuration that is requested via the information. At step 950 a compatible physical host is determined and selected from a plurality of host platforms. **At step 960 a VM is automatically deployed onto the compatible physical host platform by copying the staged VM to the compatible physical host platform**. The **automatically deploying includes installing an additional application** in the deployed VM if the additional application being requested via the information and the additional application not pre-installed in the staged VM.

Khandekar teaches autonomic deployment of images because, as explained above, it teaches an automated deployment that does not involve human intervention. *See e.g.*, Ex. 1004, 18:56-19:2 ("Since the applications are installed by the provisioning engine 810 during deployment phase (see below), they are installed unattended, that is, without any human intervention."), 5:25-36 ("The invention enables users to configure, deploy and get access to substantially 'customized' computer systems … with no need for human intervention."), 4:36-41 (VMs are "automatically prepared for deployment.").

157. Khandekar provides additional details for deploying image instances of pre-built, custom, and instantly-deployable VMs; each is sufficient by itself to render this limitation obvious. *See* Ex. 1004, 9:10-25; *supra* §VI.A.3[1e]. Khandekar's general teachings for deploying VMs apply to all three approaches; this would have been obvious to a POSITA in view of the organization of Khandekar's specification and teachings, which present general concepts related to automated deployment and more specific details regarding how those concepts apply to the particulars of pre-built, custom, and instantly-deployable VMs.

158. **(1) Pre-Built, Staged VMs.** Khandekar teaches copying pre-built VMs, with operating system and software applications pre-installed, from the VM staging repository (library 830) to host computers that are automatically selected by a heuristics engine. *See* Ex. 1004, 9:15-25, 16:42-63, 15:40-16:12, 10:15-26; *supra* §VI.A.3[1e]. For staged VMs, Khandekar teaches that "'[d]eployment' is the step of taking a staged VM, ***copying*** it to a host machine and letting it configure with a new identity." Ex. 1004, 8:64-67 (emphasis added), 9:1-3. Khandekar teaches that VMs are deployed "from a library 830 of staged VMs" to a host machine that the heuristics engine 880 determines is most appropriate. *See id.*, 10:27-42 (emphasis added):

> A heuristics engine 880 within the provisioning server then takes this user input and applies heuristics/rules to select the most appropriate VM ***from a library 830 of staged VMs***. At least one host computer

1000, 1001 is made available to the provisioning server, and ***the heuristics engine 880 also selects which host machine 1000, 1001 is most appropriate to provision and deploy the VM on***. …

159. Knowledge-based systems for autonomically deploying applications were well-known in the art. Khandekar explains that "[t]he system designer will be able to determine which heuristics/rules to implement in the heuristics engine 880 using known design criteria. Known expert systems, 'intelligent assistants,' and other knowledge-based systems and engines that can be embedded in other products and loaded with domain knowledge may be included in the heuristics engine 880 to implement the various decision functions." *Id.*, 10:35-42. This is consistent with my personal knowledge of the art at that time. Khandekar and the '138 patent both use rule engines that were known in the art.

160. **(2) Custom-Built VMs.** Custom-built VMs are also staged and are therefore copied to host computers as explained above for pre-built, staged VMs. Khandekar teaches that custom-built VMs may have additional applications, where installation images of those applications are copied from the applications library to the host and installed at the host. *See* Ex. 1004, 18:56-19:2 ("The applications library 850 is a repository of the software and data needed to install an application 856 in a VM."), 21:25-64 (describing the algorithm the provisioning engine 810 used for provisioning a fully configurable (custom-built) VM, including copying applications from "Application Library" to the host). Khandekar teaches that the

heuristics engine automatically determines the host to deploy the VM on, and then the VM files including its VM disk files with operating system and applications, are copied to the host. *See* Ex. 1004, 21:25-64:

> …
>
> 2) Call the heuristics engine 880, to determine the VM's hardware configuration and the host to deploy on.
>
> …
>
> 7) For each application selected by the user, create the custom answer file 864 and the application installation software 862 using the process described above in the "Application Library" section.
>
> 8) Copy the VM's config file and the disk files to the host.
>
> …

161. Additional install images for additional software applications are copied to the host during deployment, and then installed on the host at the end of the process. *See* Ex. 1004, 4:42-51 ("Once a VM has been configured and deployed, note that it will also be possible to alter its configuration, for example by upgrading the software and/or hardware, installing different applications, etc."), 23:54-24:27, 23:28-51 (emphasis added):

> At step 940, the status of a plurality of hosts platforms is monitored and the appropriate staged VM is determined and selected from the set of staged VMs. The staged VM most closely incorporates a configuration that is requested via the information. At step 950 a

> compatible physical host is determined and selected from a plurality of host platforms. At step 960 a VM is automatically deployed onto the compatible physical host platform by copying the staged VM to the compatible physical host platform. **The automatically deploying includes installing an additional application in the deployed VM if the additional application being requested via the information and the additional application not pre-installed in the staged VM.**

162. **(3) Instantly Deployable VMs.** Khandekar teaches copying fully-configured, instantly-deployable VMs, also with pre-installed operating system and software applications, from the instantly-deployable VMs repository (library 840) to host computers that are automatically selected by a heuristics engine. *See* Ex. 1004, 14:4-18 ("A brief description of the operating system, service pack and ***set of applications installed*** in each instantly deployable VM is preferably also included … the heuristics engine 880 applies the heuristics and rules to determine the best host, and provisions the VM on that host.") (emphasis added), 11:20-35 (emphasis added):

> According to an alternative "instant-deployment" embodiment of the invention, the user selects one of a plurality of fully configured VMs, which are stored in a library 840 of instant deployment VMs and are kept in a suspended state until deployed. The provisioning server then selects an appropriate host 1000, 1001 on which to provision the VM, ***copies the necessary data on that host***, and resumes the execution of the VM on the new host. … Since the host servers have to have a

hardware configuration identical to that where the VM was suspended, the heuristics engine will select the appropriate host server to deploy the VM on.

See also Ex. 1004, 21:65-22:13:

For provisioning an instantly-deployable VM, the algorithm followed in the prototype of the invention was as follows:

…

2) Call the heuristics engine to determine the host to deploy on.

…

4) Copy the VM's config file 842, REDO logs 843 for the disks 841 and suspend-to-disk image file 844 to the host.

…

163. Thus, for the reasons explained above, Khandekar renders this limitation obvious. *See* §VI.A.1.

4. Dependent Claim 2

164. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 2 obvious.

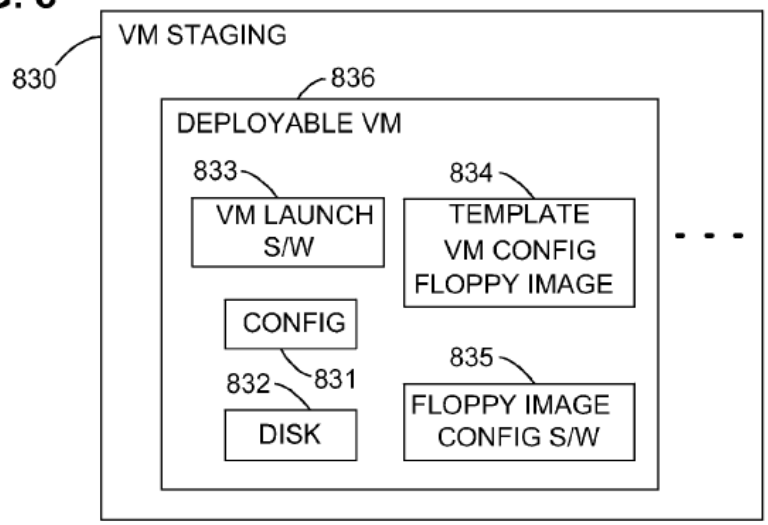
2.	**The distributed computing system of claim 1, wherein the image instances of the plurality of software applications comprise virtual disk image files.**

165. Khandekar alone (Ground 1), or in combination with Le (Ground 2), teaches the distributed computing system of claim 1. *See supra* §VI.A.3. For both

Grounds 1 and 2, Khandekar teaches or suggests the additional limitation of claim 2, wherein the image instances of the plurality of software applications comprise virtual disk image files (e.g., "VM disk file 832," "virtual disk file 841").

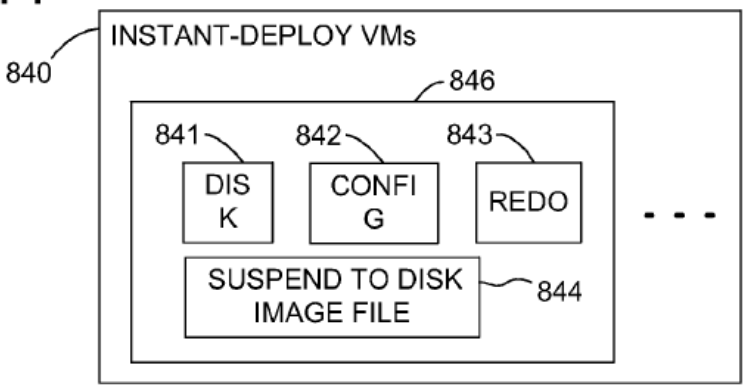
166. For staged VMs, including pre-built and custom built VMs, Khandekar teaches that the staged VMs include "VM disk files." *See* Ex. 1004, 16:47-63 ("For each VM, the data sets kept in the VM staging server 830 are illustrated in FIG. 3 and are as follows: … 2) VM disk files 832. …"), FIG. 3 (showing "DISK" 832):

FIG. 3



167. Khandekar teaches that instantly-deployable VMs include a "virtual disk file 841." Ex. 1004, 16:43-63, 17:30-50, FIG. 4:

FIG. 4



168. A POSITA would have found it obvious that Khandekar teaches the claimed virtual disk image files because the '138 patent relies on VMware software for its embodiments, and Khandekar is a VMware patent. The '138 patent states that "if virtual machine manager 402 is an instance of **VMWare ESX**, virtual machine disk image files 408 may be specially captured **.vmdk files**." Ex. 1001, 33:25-41 (emphasis added). A POSITA with knowledge of virtual machines and VMware software would have understood that the VMDK file format is short for "Virtual Machine Disk." Therefore, it would have been obvious that Khandekar's teaching of "VM disk files" and "virtual disk file" include the virtual machine disk image files used in VMware products, which are used by the embodiments of the '138 patent.

169. Thus, for the reasons explained above, Khandekar renders the limitations of Claim 2 obvious. *See* §VI.A.1.

5. Dependent Claim 3

170. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 3 obvious.

3.	**The distributed computing system of claim 1, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises an operating system.**

171. Khandekar alone (Ground 1), or in combination with Le (Ground 2), teaches the distributed computing system of claim 1. *See supra* §VI.A.3. For both Grounds 1 and 2, Khandekar teaches or suggests the additional limitation of claim 3, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises an operating system (e.g., Linux Red Hat, Windows 2000, Windows Me). *See, e.g.*, Ex. 1004, 10:15-26 ("… custom-built VMs … allow separate specification of VM components such as a hardware platform, OS, and applications."), 14:4-18 ("A brief description of the operating system, service pack and set of applications installed in each instantly deployable VM is preferably also included."), 11:36-49:

> Different choices for virtual hardware, system software, and applications may be stored in respective libraries using any known data structures. For example, one such library could contain the code of different operating systems such as Linux Red Hat, Windows 2000, Windows Me, etc., and/or different versions or releases of a single operating system, as well as other system software such as drivers, etc.

Khandekar teaches three approaches of images: pre-built, staged VMs, custom-built VMs, and instantly-deployable VMs. As explained for claim 1, each approach includes a virtual disk image with an operating system. *See supra* §VI.A.3[1e]; Ex. 1004, 9:10-25.

172. **(1) Pre-Built, Staged VMs.** Khandekar teaches images of pre-built, staged VMs stored in a repository (e.g., library 830) that include virtual disk images with operating systems. *See, e.g.*, Ex. 1004, 18:16-23 ("Staging VMs: This data structure is required for provisioning pre-built or custom-built VMs and contains … Operating system name…"), 15:40-16:12 ("… pre-built VMs will be stored with their ***operating systems and applications installed*** … In the first step, a model VM is created and the ***necessary OS and applications are installed*** in it.") (emphasis added), 5:25-36 ("…the invention makes this possible by allowing the user either to select one of a set of 'model' virtual machines that are ***pre-built*** with such components as virtual hardware, ***an operating system*** and one or more applications, or to specify desired components that are then assembled into a virtual machine….") (emphasis added), 8:50-55, 10:15-29, 14:19-25, 16:43-52, FIG. 3; *supra* §VI.A.3[1e].

173. **(2) Custom-Built VMs.** Custom-built VMs include a virtual disk with the operating system installed. *See* Ex. 1004, 15:41-46 ("…custom-built VMs will be stored with just their operating systems installed."), 15:41-49; *supra*

§VI.A.3[1e]. Custom-built VMs may have additional applications installed, which are stored in an applications library 850. *See e.g.*, Ex. 1004, 10:27-42 ("The heuristics engine 880 also selects the applications from a library 850 of available applications to install in the newly created VM."), 12:13-23 ("… to store in the library 850 different copies of applications, with different settings for the different possible operating systems, ***different copies of operating systems*** configured for different processors, etc.") (emphasis added), 18:56-19:2, 11:50-58, FIG. 5.

174. **(3) Instantly-Deployable VMs.** Khandekar teaches images of instantly-deployable VMs stored, with an operating system, in the "instantly-deployable VMs repository 840." Ex. 1004, 18:1-15 ("Instantly deployable VMs: This is a data structure indexed by the name of the VM configuration, and contains the following fields: … b) Operating system … f) Location of the VM in the instantly-deployable VMs repository 840."), 14:4-18 ("A brief description of the operating system, service pack and set of applications installed in each instantly deployable VM is preferably also included."), FIG. 4, *supra* §VI.A.3[1e]. Instantly-deployable VMs are first created according to the steps outlined for pre-built, staged VMs, which also include a virtual disk image file with an operating system. *See, e.g.*, Ex. 1004, 15:49-52 ("In the first step, a model VM is created ***and the necessary OS and applications are installed in it***.") (emphasis added), 17:30-

50 (referring to the VM staging steps for pre-built VMs), 16:43-63; *supra* §VI.A.3[1e].

175. Thus, for the reasons explained above, Khandekar renders this claim limitation obvious. *See* §VI.A.1.

6. Dependent Claim 4

176. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 4 obvious.

4.	**The distributed computing system of claim 1, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises a server application.**

177. Khandekar alone (Ground 1), or in combination with Le (Ground 2), teaches the distributed computing system of claim 1. *See supra* §VI.A.3. For both Grounds 1 and 2, Khandekar teaches or suggests the additional limitation of claim 4, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises a server application. For example, Khandekar teaches a first image instance of the plurality of image instances of the plurality of software applications comprises a server application (e.g., web server, database server). *See* Ex. 1004, 11:50-58 ("… library 850 … contain[s] code for any collection of the myriad of existing applications, from web servers, database servers, spreadsheets and word-processing programs, to games and browsers."), 12:24-39 ("For example, a user could indicate that he wants the system according

to the invention to build a (virtual) web server running a Window OS"). Therefore, Khandekar's teachings satisfy this claim limitation.

178. This is confirmed by the '138 patent, which includes an embodiment where the server application is a web server (Microsoft Internet Information Server). *See* Ex. 1001, 33:11-24:

> For example, if virtual machine 404A emulates an x86 machine, an instance of the Microsoft Windows operating system (e.g., OS 406A) may execute on virtual machine 404A as through the instance of Windows were running on a physical x86 machine. ***Other software applications*** may be installed and configured on the guest operating system running on virtual machine 404. ***For example, an instance of Microsoft Internet Information Server*** may execute on an instance of Microsoft Windows that is executing on a virtual machine that emulates an x86 machine.

Therefore, the specification of the '138 patent confirms that a web server satisfies the claimed server application.

179. Khandekar teaches, e.g., a web server as one of the software applications for the plurality of image instances of the plurality of software applications. Thus, for the reasons explained above, Khandekar by itself renders this claim limitation obvious. *See* §VI.A.1.

7. Dependent Claim 5

180. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 5 obvious.

5.	**The distributed computing system of claim 1, wherein a first one of the virtual machines provides an environment that emulates an x86 platform.**

181. Khandekar alone (Ground 1), or in combination with Le (Ground 2), teaches the distributed computing system of claim 1. *See supra* §VI.A.3. For both Grounds 1 and 2, Khandekar teaches or suggests the additional limitation of claim 5, wherein a first one of the virtual machines provides an environment that emulates an x86 platform.

182. For example, as discussed in Claim 3, Khandekar teaches that the image instances of the software applications comprise an operating system including Linux Red Hat, Windows 2000, Windows Me. *See supra* §VI.A.5, Ex. 1004, 11:36-49. A POSITA would have understood that these operating systems typically ran on x86 platforms—indeed, x86 was the primary target platform for Windows 2000 and Windows Me. The x86 platform was the predominant computing platform at the time, and x86 processors were provided by major chip makers such as Intel with its Pentium processors and AMD. VMware's software, including its ESX and GSX software, provided an emulation of x86 platform. *See, e.g.*, 1004 13:17-35 ("Several products are available from

VMware"). Thus, it would have been obvious one of the virtual machines provides an environment that emulates an x86 platform because the x86 platform was common and well-know at the time, particularly for the operating systems taught by Khandekar including Linux Red Hat, Windows 2000, Windows Me. *See, e.g.*, Ex. 1004, 11:36-49.

183. Thus, for the reasons explained above, Khandekar renders the limitations of Claim 5 obvious. *See* §VI.A.1.

Ground 2: Le's teachings, when combined with Khandekar, further render this claim obvious

184. Le provides additional teachings that complement and elaborate on Khandekar's teachings, further rendering this limitation obvious. *See* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le).

185. For example, Le teaches or suggests the distributed computing system of claim 1, *see supra* §VI.A.3, wherein a first one of the virtual machines provides an environment that emulates an x86 platform. For example, Le teaches that the VMM in its virtualized computer systems supports x86 virtual machines as described in Bugnion, which it incorporates by reference. *See* Ex. 1005, 12:63-13:11 (citing and incorporating Bugnion by reference); Ex. 1012, 18:28-38 ("the VMM is capable of running all major operating systems (Microsoft DOS,

Windows 3.1, Windows 95, Windows 98, Windows NT, Linux, and Unix) for the Intel x86 architecture in a virtual machine.").

186. A POSITA would have found it obvious and been motivated to apply Le's teachings (which incorporate Bugnion by reference) because x86 platforms were the most common computer platform, and virtual machines commonly emulated an x86 platform. Therefore, the combined teachings of Khandekar and Le further render this claim obvious. *See* §VI.A.2.

8. Dependent Claim 9

187. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 9 obvious.

9.	**The distributed computing system of claim 1, wherein the control node further comprises one or more rule engines that provide autonomic deployment of the software applications to the virtual machines in accordance with a set of one or more rules.**

188. Khandekar alone (Ground 1), or in combination with Le (Ground 2), teaches the distributed computing system of claim 1. *See supra* §VI.A.3. For both Grounds 1 and 2, Khandekar teaches or suggests the additional limitation of claim 9, wherein the control node (e.g., "provisioning server") further comprises one or more rule engines (e.g., heuristics/rules engine) that provide autonomic deployment of the software applications to the virtual machines in accordance with a set of one or more rules. *See, e.g.*, Ex. 1004, 10:27-42 (emphasis added):

> A heuristics engine 880 within the provisioning server then takes this user input and applies heuristics/rules to select the most appropriate VM from a library 830 of staged VMs. At least one host computer 1000, 1001 is made available to the provisioning server, and ***the heuristics engine 880 also selects which host machine 1000, 1001 is most appropriate to provision and deploy the VM on.*** The heuristics engine 880 also selects the applications from a library 850 of available applications to install in the newly created VM. The system designer will be able to determine which heuristics/rules to implement in the heuristics engine 880 using known design criteria. Known expert systems, "intelligent assistants," and other **knowledge-based systems and engines** that can be embedded in other products and loaded with domain knowledge may be included in the heuristics engine 880 to implement the various decision functions.

Khandekar teaches the use of rules and "knowledge-based systems and engines," which were known in the art to include rules engines. *See id.* Khandekar also teaches that a system designer would have been able to determine which "rules to implement in the heuristics engine 880 using known design criteria." *Id.* Thus, Khandekar's rules engine provides autonomic deployment of the software applications to the virtual machines in accordance with a set of one or more rules. *See id.* The particular rules are selected by the system designer. *See id.*

189. The '138 patent does not purport to invent a rules engine and instead relies on rules engines that were known in the art, including the RETE, TREAT,

and LEAPS algorithms. *See, e.g.*, Ex. 1001, 26:45-61 (citing various prior art references from the 1980s and 1990s). Khandekar teaches the use of known rules engines; therefore, since RETE, TREAT, and LEAPS were known rules engines, Khandekar's teachings render this claim limitation obvious.

190. Khandekar teaches that the heuristics engine provides autonomic deployment according to heuristics/rules based on the state of the hosts. *See* Ex. 1004, 12:57-13:8:

> Information about the current state of the different hosts is preferably input dynamically into a database or other data structure in the heuristics engine 880. Host state information can then be used to suggest components, as well as the properties of VMs that need to be matched with user requests and any static rules pre-defined by the system administrator. The heuristics engine may then also efficiently direct deployment and migration in order to balance the load on the different available hosts, for example by deploying a selected, staged VM to the host whose processor currently has the most available cycles.

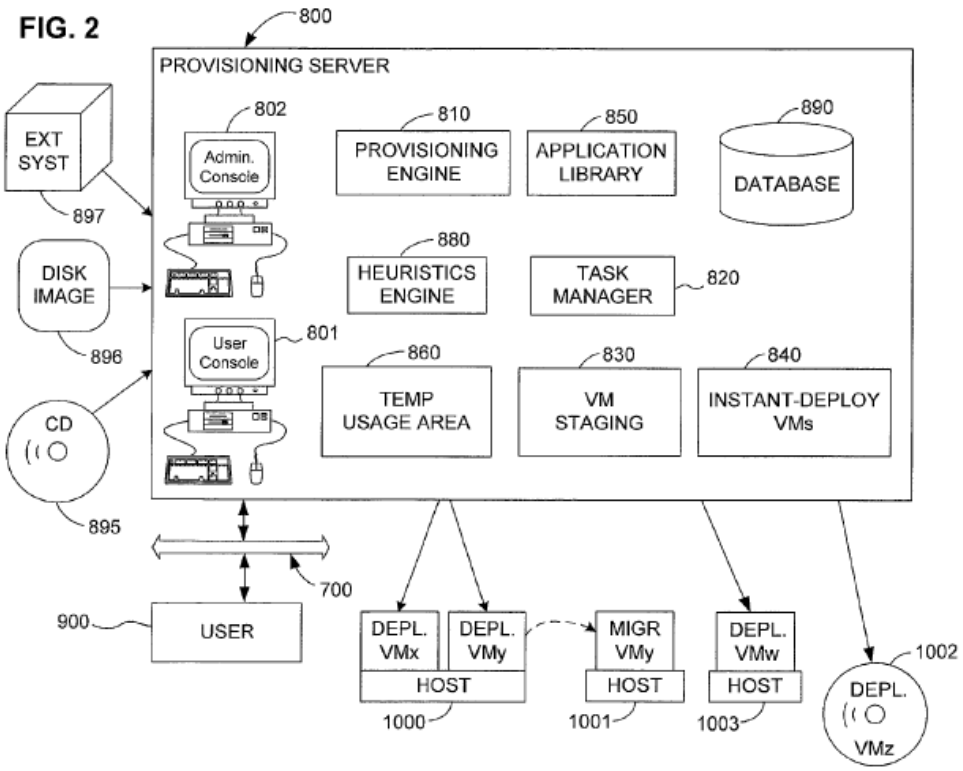
See also 4:58-65 ("… the status of the hosts is monitored and the host on which a configured VM is to be deployed is selected heuristically. A deployed VM may also be migrated from one host to another based on the monitored status of the plurality of hosts."), 23:54-24:27. As an example, Khandekar teaches monitoring whether a host is overloaded, e.g., if the host load crosses a threshold. *See*

Ex. 1004, 13:44-56 ("Hosts can be monitored for load and ***if the load crosses a threshold*** …") (emphasis added). Le also teaches monitoring status and performance information. *See* Ex. 1005, 68:21-24 ("… the agent 7300 can also report status and system performance information to the imaging server 2101, allowing the UCMS to detect the presence of the agent and to monitor the deployed computer's health.").

191. These teaching are also consistent with the specification of the '138 patent, which similarly looks to whether a server status is "overloaded" and then takes action accordingly with its rules engine. *See, e.g.*, Ex. 1001, 25:60-26:25. Therefore, Khandekar renders this limitation obvious.

192. Khandekar explains how its rules engine is used to provide autonomic deployment of the software applications to the virtual machines on the host computers. *See* Ex. 1004, 20:31-62 ("… the heuristics/rules engine 880, … matches the information against the database 890 to select the appropriate host and the VM hardware configuration. … the heuristics engine … [s]can[s] the list of available host machines 1000, 1001, etc., and determine[s] the one with enough memory and disk space to host the VM about to be created. …"), 11:11-19 ("… the heuristics engine 880 suggests the best available match by applying a database of heuristic rules about properties of VMs, available applications and operating systems, virtualized devices, etc."), 12:24-39 ("… the heuristics engine 880 will

pass parameters to other components such that they will build a VM having the desired characteristics."), 14:4-18 ("When the user selects a configuration, the heuristics engine 880 applies the heuristics and rules to determine the best host, and provisions the VM on that host."), 14:35-49 ("… one of the heuristics and rules included in the engine 880 may be a database of the minimum and recommended amounts of RAM for the different OSs and applications available for provisioning."), FIG 2:



193. Thus, for the reasons explained above, Khandekar renders this claim limitation obvious. *See* §VI.A.1.

9. Independent Claim 19

194. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 19 obvious.

19[a]. A method comprising:

195. I understand that a preamble generally does not state a claim limitation. However, to the extent that Patent Owner argues that the preamble states a limitation, it is my opinion that it is taught by Khandekar as explained below.

196. For example, Khandekar teaches "a method and system implementation for creating a virtualized computer system …" Ex. 1004, 4:30-35; *see also id.*, 23:1-20, 4:30-35, 23:26-51 ("FIG. 7 illustrates a method 900 of creating a virtualized computer system."), FIG. 7. Khandekar teaches various steps and algorithms performed by its provisioning server, as explained previously for claim 1. *See supra* §VI.A.3.

197. Therefore, Khandekar renders element [19a] obvious. *See* §VI.A.1.

Ground 2: Le's teachings, when combined with Khandekar, further render this claim element obvious.

198. Le provides additional teachings that complement and elaborate on Khandekar's teachings, further rendering this limitation obvious. *See* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le).

199. For example, Le teaches an "Image Capture and Deployment Method." *See, e.g.*, Ex. 1005, 59:21-37, 64:59, 65:21-33. Le teaches various steps and algorithms, as explained for claim 1. *See supra* §VI.A.3. Therefore, Le's additional teachings further render element [19a] obvious. *See* §VI.A.2.

[19b]	**storing a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes in a distributed computing system,**

200. As an initial matter, limitation [19b] is similar to limitation [1c], which recites the same "plurality of image instances…" as shown in green below:

Limitation [19b]	Limitation [1c]
storing a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes in a distributed computing system	a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes

Khandekar alone (Ground 1), and in combination with Le (Ground 2), teaches limitation [1c]. *See supra* §VI.A.3[1c]. That analysis applies here and therefore renders this shared language obvious.

201. Limitation [19b] also recites "storing" the plurality of image instances "in a distributed computing system." Khandekar alone (Ground 1), and in combination with Le (Ground 2), render this obvious by teaching, as explained above, a distributed computing system (*see supra* §VI.A.3[1a]) comprising a software image repository comprising non-transitory, computer-readable media

operable to store the plurality of image instances recited in [1c] (*see supra* §VI.A.3[1b]). That analysis applies here. Therefore, since the software image repository was operable to store them, it would have been obvious to store the plurality of image instances in the repository of the distributed computing system. As explained above, Khandekar teaches or suggests a provisioning server that is a software image repository for storing (i) a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes, and (ii) a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines. *See supra* §VI.A.3[1b], [1c], [1e].

202. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render this limitation obvious. *See* §§VI.A.1, VI.A.2.

[19c]	**wherein the image instances of the virtual machine manager provide a plurality of virtual machines, each virtual machine operable to provide an environment that emulates a computer platform;**

203. Limitation [19c] is identical to limitation [1d] except [19c] omits "when executed on the applications nodes." This omission does not materially alter my analysis because, while [19c] might be broader than [1d], it is satisfied by the teachings of Khandekar alone (Ground 1), and in combination with Le (Ground 2), that render [1d] obvious. The analysis from limitation [1d] applies here. *See supra* §VI.A.3[1d]. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render this limitation obvious. *See* §§VI.A.1, VI.A.2.

[19d] storing a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines;

204. Limitation [19d] is identical to limitation [1e] except [19d] further requires "storing." Khandekar by itself (Ground 1), and in combination with Le (Ground 2), teach limitation [1e], and that analysis applies here. *See supra* §VI.A.3[1e].

205. Regarding "storing," that limitation is taught by Khandekar alone (Ground 1), and in combination with Le (Ground 2), which as explained above for limitation [1b], both teach a software image repository comprising non-transitory, computer-readable media operable to store the plurality of image instances recited in limitation [1e]. *See supra* §VI.A.3[1b], [1e]. That analysis applies here. Therefore, since the software image repository was operable to store them, it would have been obvious to store the plurality of image instances, as described for [19d], in the repository of the distributed computing system.

206. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render this limitation obvious. *See* §§VI.A.1, VI.A.2.

[19e] executing a rules engine to perform operations to autonomically deploy the image instances of the virtual machine manager on the application nodes by causing the image instances of the virtual machine manager to be copied to the application nodes; and

207. Limitation [19e] is similar to limitation [1f], which recites the latter portion of [19e] starting from "the image instances…" as shown in green below:

Limitation [19e]	Limitation [1f]
executing a rules engine to perform operations to autonomically deploy **the image instances of the virtual machine manager on the application nodes by causing the image instances of the virtual machine manager to be copied to the application nodes**	a control node that comprises an automation infrastructure to provide autonomic deployment of the plurality of **image instances of the virtual machine manager on the application nodes by causing the** plurality of **image instances of the virtual machine manager to be copied** from the software image repository **to the application nodes**

208. Khandekar alone (Ground 1), and in combination with Le (Ground 2), teaches limitation [1f]. *See supra* §VI.A.3[1f]. That analysis applies here and therefore renders this shared language obvious. The omission of "plurality of" and "from the software image repository" does not materially alter my analysis because, while this part of [19e] might be broader than the corresponding part of [1f], it is still satisfied by the teachings that render [1f] obvious.

209. Regarding the remainder of [19e], Khandekar teaches or suggests a control node that comprises an automation infrastructure (e.g., "heuristics/rules engine") to *provide autonomic deployment* of the plurality of image instances of the virtual machine manager on the application nodes, as discussed for limitation [1f]. *See supra* §VI.A.3[1f]. That analysis applies here. Therefore, Khandekar renders obvious the step of executing a rules engine (e.g., the "heuristics/rules engine") to perform operations to autonomically deploy the "image instances…" recited in the remainder of the claim. *See id.*

210. Thus, for the reasons explained above, Khandekar renders this limitation obvious. *See* §VI.A.1.

[19f]	**executing the rules engine to perform operations to autonomically deploy the image instances of the software applications on the virtual machines by causing the image instances of the software applications to be copied to the application nodes.**

211. Limitation [19f] is similar to limitation [1g], which recites the latter portion of [19f] starting from "the image instances…" as shown in green below:

Limitation [19f]	**Limitation [1g]**
executing a rules engine to perform operations to autonomically deploy **the image instances of the software applications on the virtual machines by causing the image instances of the software applications to be copied to the application nodes**	to provide autonomic deployment of **the** plurality of **image instances of the software applications on the virtual machines by causing the plurality of image instances of the software applications to be copied** from the software image repository **to the application nodes**

212. Khandekar alone (Ground 1), and in combination with Le (Ground 2), teaches limitation [1g]. *See supra* §VI.A.3[1f]. That analysis applies here and therefore renders this shared language obvious. The omission of "plurality of" and "from the software image repository" does not materially alter my analysis because, while this part of [19f] might be broader than the corresponding part of [1g], it is still satisfied by the teachings that render [1g] obvious.

213. Regarding the remainder of [19f], Khandekar teaches or suggests a control node that comprises an automation infrastructure (e.g., "heuristics/rules engine") to *provide autonomic deployment* of the plurality of image instances of the software applications on the virtual machines. *See supra* §VI.A.3[1g]; *see also supra* §VI.A.8[9] (Khandekar teaches or suggests "one or more rule engines that provide autonomic deployment of the software applications to the virtual machines"). That analysis applies here. Therefore, Khandekar renders obvious the step of executing the rules engine (e.g., the "heuristics/rules engine") to perform operations to autonomically deploy the "image instances…" recited in the remainder of the claim. *See id.*

214. Thus, for the reasons explained above, Khandekar renders this limitation obvious. *See* §VI.A.1.

10. Dependent Claim 20

215. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 20 obvious.

20.	**The method of claim 19, wherein storing the image instances of the plurality of software applications comprises storing virtual disk image files.**

216. Claim 20 is nearly identical to claim 2, except it additionally recites language for "storing." The "storing" limitation is taught by Khandekar alone (Ground 1), and in combination with Le (Ground 2), which as explained above

teach or suggest the method of claim 19, including the step of *storing* the plurality of image instances of a plurality of software applications. *See supra* §VI.A.9[19d], §VI.A.3[1b].

217. For both Grounds 1 and 2, Khandekar teaches or suggests the additional limitation of claim 20, wherein storing the image instances of the plurality of software applications comprises storing virtual disk image files. As explained above for claim 2, Khandekar teaches or suggests that the image instances of the plurality of software applications comprise virtual disk image files. *See supra* §VI.A.4. That analysis applies here and renders this limitation obvious.

218. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render Claim 20 obvious. *See* §§VI.A.1, VI.A.2.

11. Dependent Claim 21

219. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 21 obvious.

21.	**The method of claim 19, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises an operating system.**

220. Khandekar by itself (Ground 1), and in combination with Le (Ground 2), teach or suggest the method of claim 19. *See supra* §VI.A.9. The limitations of Claim 21 are identical to those recited for Claim 3. That analysis applies here. Thus, Claim 21 is rendered obvious by Khandekar by itself, and in combination

with Le, for at least the reasons explained previously for Claim 3. *See supra* §VI.A.5.

12. Independent Claim 26

221. It is my opinion that Khandekar by itself, and the combination of Khandekar and Le, render the limitations of Claim 26 obvious.

26[a]. A non-transitory, computer-readable medium comprising instructions stored thereon, that when executed by a programmable processor:

222. I understand that a preamble generally does not state a claim limitation. However, to the extent that Patent Owner argues that the preamble states a limitation, it is my opinion that it is taught by Khandekar as explained below.

223. Khandekar teaches or suggests a non-transitory, computer-readable medium comprising instructions stored thereon that are executed by a programmable processor. *See, e.g.,* Ex. 1004, 9:26-45 ("The main feature of the invention is a VM provisioning server 800, which, like any conventional computer system, will include system hardware, system software, devices, etc. as needed these are standard components of any computer and are therefore not described in further detail."). Khandekar teaches that its provisioning server includes hardware and software for implementing its features, and therefore it would have been obvious to use standard computer components, such as a processor, to execute

software (instructions) that implements the functions of the provisioning server. *See id.*

224. It would also have been obvious to use a non-transitory, computer-readable medium, such as a hard disk, to store the provisioning server's software because hard disks were included in the vast majority of personal computers and servers in the 2006 timeframe and therefore would have been among the "standard components" that Khandekar teaches for its provisioning server. *See* Ex. 1004, 9:26-45. Most computers used a hard disk to store software that ran on one or more processors. Thus, for the reasons explained above, Khandekar renders this limitation obvious. *See* §VI.A.1.

Ground 2: Le's teachings, when combined with Khandekar, further render this claim element obvious.

225. Le provides additional teachings that complement and elaborate on Khandekar's teachings, further rendering this limitation obvious. *See* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le).

226. Le teaches or suggests a non-transitory, computer-readable medium comprising instructions stored thereon that are executed by a processor. For example, Le teaches using software installed on a computer to implement the features of its UCMS, which plays an analogous role to Khandekar's provisioning server (*see supra* §VI.A.2), with further teachings that software programs comprise

computer-executable instructions and are stored on disks in the server. *See, e.g.*, Ex. 1005, 59:29-36 ("The core component of the UCMS is the UCMS server software 2101, which is installed on a server computer 2000… [and] manages the files used by the UCMS, maintains several databases that keep track of the files and managed computers and coordinates all disk imaging tasks, such as capture, deployment, reconfiguration and customization."), 60:13-18 ("The imaging server uses multiple databases to provide permanent and stable storage for the data critical to the operation of the UCMS."), 23:44-52 ("The imaging server 2101 itself is an application …"), 89:8-31 ("… the imaging server program comprising computer-executable instructions …"), 26:38-52 (application files are "stored on the server disk 2010" and "are loaded into memory when the server computer is booted."), FIG. 4.

227. Based on the combined teachings of Khandekar and Le, a POSITA would have found it obvious to use a non-transitory, computer-readable medium, such as a hard disk, to store instructions that when executed by a programmable processor, implement the features of the provisioning server, as explained above. Therefore, Le's additional teachings further render element [1a] obvious. *See* §VI.A.2.

[26b] store a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes in a distributed computing system,

228. Limitation [26b] is identical to limitation [19b], except [26b] recites "store" while [19b] recites "storing." This difference in verb tense does not materially alter my analysis. Khandekar by itself (Ground 1), and in combination with Le (Ground 2), render limitation [19b] obvious, and that analysis applies here. *See supra* §VI.A.9[19b]. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render this limitation obvious. *See* §§VI.A.1, VI.A.2.

[26c] wherein the image instances of the virtual machine manager provide a plurality of virtual machines, each virtual machine operable to provide an environment that emulates a computer platform;

229. Limitation [26c] is identical to limitation [19c]. Thus, limitation [26c] is rendered obvious by Khandekar by itself (Ground 1), and in combination with Le (Ground 2), for at least the reasons explained previously for limitation [19c]. *See supra* §VI.A.9[19c].

[26d] store a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines;

230. Limitation [26d] is identical to limitation [19d] except [26d] recites "store" while [19d] recites "storing." This difference in verb tense does not materially alter my analysis. Khandekar by itself (Ground 1), and in combination

with Le (Ground 2), renders limitation [19d] obvious, and that analysis applies here. *See supra* §VI.A.9[19d]. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render this limitation obvious. *See* §§VI.A.1, VI.A.2.

[26e]	**perform operations to autonomically deploy the image instances of the virtual machine manager on the application nodes by causing the image instances of the virtual machine manager to be copied to the application nodes; and**

231. Limitation [26e] is identical to limitation [19e] except [26e] omits "executing a rules engine to." This omission does not materially alter my analysis because, while [26e] might be broader than [19e], it is still satisfied by the teachings that render [19e] obvious. Khandekar by itself (Ground 1), and in combination with Le (Ground 2), render limitation [19e] obvious, and that analysis applies here. *See supra* §VI.A.9[19e]. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render this limitation obvious. *See* §§VI.A.1, VI.A.2.

[26f]	**perform operations to autonomically deploy the image instances of the software applications on the virtual machines by causing the image instances of the software applications to be copied to the application nodes.**

232. Limitation [26f] is identical to limitation [19f] except [26f] omits "executing a rules engine to." This omission does not materially alter my analysis because, while [26f] might be broader than [19f], it is still satisfied by the

teachings that render [19f] obvious. Khandekar by itself (Ground 1), and in combination with Le (Ground 2), render limitation [19f] obvious, and that analysis applies here. *See supra* §VI.A.9[19f]. Thus, for the reasons explained above, Khandekar by itself, and in combination with Le, render this limitation obvious. *See* §§VI.A.1, VI.A.2.

B. Ground 3: Claims 9 and 10 are Rendered Obvious by Khandekar in View of Le and Sidebottom

233. It is my opinion that a POSITA would have found Claims 9 and 10 obvious based on the combined teachings of Khandekar, Le, and Sidebottom.

1. Motivation to Combine Khandekar, Le, and Sidebottom

234. A POSITA would have been motivated to combine the teachings of Khandekar, Le, and Sidebottom. The combination of Khandekar and Le was explained for Ground 2, and a POSITA would have been further motivated to apply Sidebottom's teachings regarding a rules engine with a forward-chaining algorithm, as explained below. *See, e.g.*, Ex. 1006, Abstract, 1:36-44.

235. The combination of Khandekar and Le teaches or suggests a repository for VM and VMM images with a heuristics/rules engine that automatically deploys those images to host computers based on the monitored status of the hosts. *See, e.g.*, Ex. 1004, Abstract, 4:30-35, 4:58-65, 10:27-43, 13:17-35; Ex. 1005, 59:29-36, 68:21-24; *supra* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le's teachings), §VI.A.3[1f]-[1g],

§VI.A.8. Khandekar teaches a heuristics/rules engine using rules and knowledge-based systems that were known in the art to select appropriate VMs, hosts, and applications. *See* Ex. 1004, 10:27-42 ("… The system designer will be able to determine which heuristics/rules to implement in the heuristics engine 880 using known design criteria. Known expert systems, 'intelligent assistants,' and other knowledge-based systems and engines that can be embedded in other products and loaded with domain knowledge may be included in the heuristics engine 880 to implement the various decision functions."). Therefore, Khandekar provides express motivation to apply known "knowledge-based systems and engines" to its teachings of a heuristics/rules engine. *See id.*

236. Sidebottom provides such a teaching in the form of a "rules engine" that applies a "forward-chaining rule-based algorithm, such as LEAP or RETE." *See, e.g.*, Ex. 1006, 3:44-58, 9:36-45. LEAP and RETE were well-known rules engine algorithms. A POSITA would have found it obvious and been motivated to apply Sidebottom's teachings to the combination because Khandekar provides express motivation to combine its teachings with known knowledge-based systems and engines, which would have included the forward-chaining rule algorithms taught by Sidebottom.

237. In addition, the prior art provides motivation to apply knowledge-based systems and engines, including those taught by Sidebottom, for monitoring

and managing computer resources, as taught by the combination of Khandekar and Le. For example, prior art reference Das teaches monitoring "facts" such as "an average load on a CPU" and applying hierarchies, or trees of rules built through "Chaining of Rules." *See* Ex. 1013 ¶¶[0083]-[0084], [0163]-[0164]. Similarly, Khandekar teaches that "[h]osts can be monitored for load…" *See* Ex. 1004, 13:44-56; *supra* §VI.A.8. Therefore, Das provides a teaching, suggestion, and motivation to apply chaining rules, such as a forward-chaining rules engine as taught by Sidebottom, in response to the monitored load of hosts, as taught by Khandekar.

238. As another example, prior art reference Russell teaches that a forward-chaining algorithm that apply Modus Ponens inference rule, which is useful for a knowledge base that make inferences in response to newly arrived information. *See* Ex. 1010 at 280. Russell teaches that forward-chaining algorithms are "used in production [i.e., condition-action rule] systems, which perform efficient updates with very large rule sets." Ex. 1010 at 310, 286 n.4. RETE algorithms have been a key component of production systems, which were among the earliest forward chaining systems in widespread use. *See* Ex. 1010 at 286. Khandekar teaches taking actions, such as migrating VMs, when a monitored load crosses a threshold (newly arrived information). *See* Ex. 1004, 13:44-56. Therefore, Russell provides a teaching, suggestion, and motivation to apply

forward-chaining rules to Khandekar, including the forward-chaining rules such as RETE as taught by Sidebottom.

239. In summary, the prior art provides teachings and suggestions that would have motivated a POSITA to combine the teachings of Khandekar, Le, and Sidebottom by applying Sidebottom's teachings regarding the use and application of forward-chaining rules engines. As explained above, forward-chaining rules engines were well-known in the art for implementing the knowledge-based systems taught by Khandekar. Indeed, the two main types of such rules engines were forward-chaining and backwards-chaining rules engines. Forward-chaining rules engines commonly used in the art to respond to monitored events, as explained above, and since Khandekar teaches that application, a POSITA would have found it obvious to apply a forward-chaining rules engine to Khandekar's teachings. At the very least, a POSITA would have found it obvious to try forward-chaining rules engine as one of a finite number of identified, predictable solutions.

240. A POSITA would also have been motivated to combine Sidebottom with Khandekar and Le because Sidebottom explains that its teachings are broadly applicable for defining policies in a computer network. *See* Ex. 1006, 1:36-38 ("In general, principles of the invention are directed to techniques to define and deploy business policies in a computer network."). Sidebottom teaches monitoring events in the network, evaluating if the condition of a condition/action rule is satisfied

using a rules engine, and if so, implementing the action. *See* Ex. 1006, 2:65-3:17, 3:32-43, 5:63-6:17, 4:26-44. As was typical for rules engines, Sidebottom teaches that its rules were "specified in terms of high-level business information." *See id.*, 1:41-44 ("The business logic rules may be defined as condition/action rules specified in terms of high-level business information that 'fire' in response to one or more low-level network events."). Therefore, Sidebottom's high-level rules were adaptable to a wide variety of applications and business needs, which was typical for rules engines in the art.

241. A POSITA would have found it obvious that a forward-chaining rules engine, such as LEAP or RETE, could be readily utilized in the combination of Khandekar and Le. Forward-chaining rules engines are general algorithms, not tied to any particular implementation, that a POSITA would have found obvious to apply to Khandekar and Le, as explained above. Furthermore, Sidebottom and Khandekar both teach the use of rules for defining condition/action rules based on monitored events in a network. *See* Ex. 1004, 10:27-4, 13:44-56 (teaching monitoring the status, such as the load, of a hosts and applying rules in a heuristics/rules engine to implement various decision functions); *See* Ex. 1006, 3:32-43, 5:63-6:17, 4:26-44 (teaching monitoring network events, evaluating the monitored events using a rules engine with a forward-chaining algorithm and implementing actions if the rule is satisfied). Therefore, applying Sidebottom's

teachings to the combination of Khandekar and Le would have used Sidebottom's rules engine teachings in a natural and conventional manner. A POSITA would have found it obvious and been motivated to combine Sidebottom's teachings with Khandekar and Le.

242. A POSITA would have had a reasonable expectation of success in combining the teachings of Khandekar, Le, and Sidebottom given the similarities in their teachings, which are directed to complimentary aspects of distributed computing systems. The combination of Khandekar and Le was explained above for Ground 2, and the additional teachings of Sidebottom specify the algorithms used by Khandekar's heuristics/rules engine. Sidebottom's forward-chaining rules algorithms, such as LEAP or RETE, were known in the art, and their application to Khandekar's heuristics/rules engine would have been for their known and conventional purpose, as explained above. Each of the respective teachings of the references are used in this combination for their known purpose, in the conventional manner in which they were known in the art and taught in the references. Khandekar teaches the use of known knowledge-based systems and engines and Sidebottom teaches such a rules engine. Distributed computing systems, virtual machines and related monitoring techniques were well known at the time. *See supra* §I. Given a POSITA's knowledge of distributed computer

systems, related monitoring techniques, and rules engines, the combination would have been well within the level of ordinary skill in the art. *See supra* §IV.

243. In addition, a POSITA would have had been motivated and had a reasonable expectation of success in combining Khandekar, Le, and Sidebottom because the combination applies known techniques (e.g., forward-chaining rules algorithms, such as LEAP or RETE) to improve similar devices (e.g., distributed computing systems in a network) in the same way as they were used in the prior art.

2. Dependent Claim 9

244. It is my opinion that the combination of Khandekar, Le and Sidebottom renders the limitations of Claim 9 obvious.

9.	**The distributed computing system of claim 1, wherein the control node further comprises one or more rule engines that provide autonomic deployment of the software applications to the virtual machines in accordance with a set of one or more rules.**

245. The combination of Khandekar and Le teaches or suggests the distributed computing system of claim 1. *See supra* §VI.A.3. Khandekar further teaches or suggests the additional rule-engine limitation of claim 9, wherein the control node (e.g., "provisioning server") further comprises one or more rule engines (e.g., "heuristics/rules engine") that provide autonomic deployment of the software applications to the virtual machines in accordance with a set of one or more rules. *See supra* §VI.A.8.

246. Sidebottom's teachings further render this limitation of claim 9 obvious (Ground 3). Khandekar teaches the use of known "knowledge-based systems and engines" (*see* Ex. 1004, 10:27-42), which Sidebottom teaches in the form of a forward-chaining rules engine, including the LEAP and RETE algorithms. Thus, a POSITA would have been motivated to apply Sidebottom's teachings to Khandekar's heuristics/rules engine. *See supra* §VI.B.1 (explaining how and why a POSITA would have combined the teachings of Khandekar, Le, and Sidebottom).

247. Sidebottom teaches a rules engine using "forward-chaining rule-based algorithm, such as LEAP or RETE." *See, e.g.*, Ex. 1006, 3:44-58, 9:36-45. Sidebottom teaches high-level condition/action rules. *See id.*, 1:41-44 ("The business logic ***rules may be defined as condition/action rules specified in terms of high-level business information*** that 'fire' in response to one or more low-level network events.") (emphasis added), 9:36-45 ("Rules engine 20 begins a condition/action rule processing cycle when the rules engine detects a change in working memory 26 (80). When rules engine 20 detects a change in working memory 26, rules engine 20 evaluates the expressions described in the 'condition' attributes of condition/action rules (82). For example, rules engine 20 may use a forward-chaining algorithm to evaluate the 'condition' attributes.").

248. For example, Sidebottom teaches monitoring events in the network, evaluating if the condition of a condition/action rule is satisfied using a rules engine, and if so, implementing the action. *See* Ex. 1006, 2:65-3:17 ("… service provider device 22A may send a network event to SDD 4 when subscriber 16 successfully authenticates. In addition, service provider devices 12 may send events such as user logins and logouts, bandwidth usage reports, assignments of network addresses, disconnections, unsuccessful authentication attempts, changes of user-specific data, network attack alerts, notices of performance degradations, and other network-level events."), 3:32-43 ("… SDD 4 applies a set of business logic rules to network events 13 to perform actions related to services provided by service provider 6. … Upon receiving an event, SDD 4 evaluates the condition of a condition/action rule. If SDD 4 determines that the condition is satisfied given the event, SDD 4 enables the action of the condition/action rule."), 5:63-6:17 ("Rules engine 20 evaluates a condition of a rule in set of condition/action rules 36 and enables an action associated with the rule when the condition is satisfied."), 4:26-44 ("By creating condition/action rules and appropriate pluggable software modules that implement the actions, SDD 4 allows an enterprise to efficiently deploy to a network a great diversity of business policies."), Abstract. Sidebottom teaches an example of its rules engine environment. *See id.*, FIG. 2:

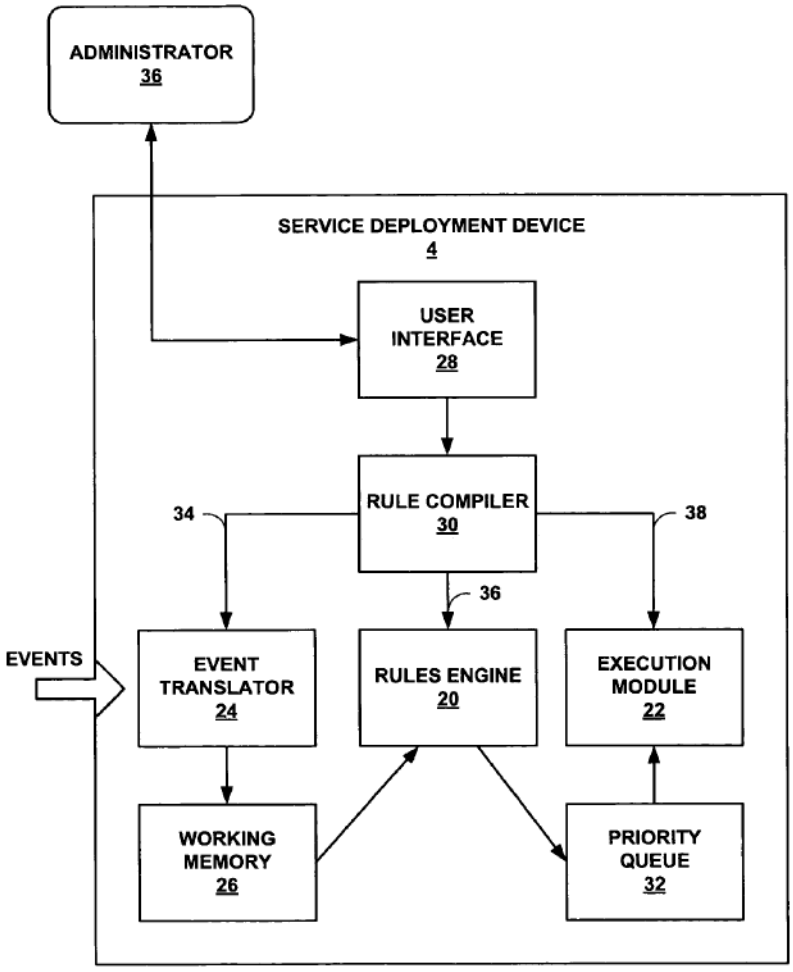


FIG. 2

249. Similarly, Khandekar similarly teaches monitoring host status, such as whether the load of a host has exceeded a threshold, and using that information for its heuristics/rules engine. *See*, *e.g.*, Ex. 1004, 4:58-65, 13:44-56 ("Hosts can be monitored for load and ***if the load crosses a threshold*** …") (emphasis added). Therefore, a POSITA would have found it obvious to apply Sidebottom's teachings to Khandekar by monitoring events in hosts—such as whether a host is overloaded—evaluating if the condition of a condition/action rule is satisfied using

a forward-chaining rules engine, and if so, implementing the action to guide the deployment of software applications. *See* Ex. 1006, 2:65-3:17, 3:32-43, 5:63-6:17, 4:26-44; *supra* §VI.A.3[1g] (explaining autonomic deployment), §VI.A.8 (same).

250. These teaching are also consistent with the specification of the '138 patent, which similarly looks to whether a server status is "overloaded" and then takes action accordingly with its rules engine. *See, e.g.*, Ex. 1001, 25:60-26:25.

251. Sidebottom's teachings complement Khandekar's teachings, and the combination renders obvious the use of one or more rule engines that provide autonomic deployment of the software applications to the virtual machines in accordance with a set of one or more rules. Thus, for the reasons explained above, the combination of Khandekar, Le and Sidebottom renders this Claim obvious. *See* §VI.B.1.

3. Dependent Claim 10

252. It is my opinion that the combination of Khandekar, Le and Sidebottom renders the limitations of Claim 10 obvious.

10.	**The distributed computing system of claim 9, wherein the one or more rule engines are forward-chaining rule engines.**

253. The combination of Khandekar, Le, and Sidebottom (Ground 3), teaches or suggests the distributed computing system of claim 9 (*see supra* §VI.B.2), wherein the one or more rule engines are forward-chaining rule engines. For example, Sidebottom teaches the use of forward-chaining rules algorithms for

rules engines. *See* Ex. 1006, 3:44-58 ("… the rule specification language allows administrator 36 to draft conditions using arithmetic, Boolean, or other operators. … In one embodiment, SDD 4 uses a forward-chaining rule-based algorithm, such as LEAP or RETE."), 9:36-45 ("Rules engine 20 begins a condition/action rule processing cycle when the rules engine detects a change in working memory 26 (80). When rules engine 20 detects a change in working memory 26, rules engine 20 evaluates the expressions described in the 'condition' attributes of condition/action rules (82). For example, rules engine 20 may use a forward-chaining algorithm to evaluate the 'condition' attributes.").

254. The '138 patent does not purport to invent a rules engine and instead relies on rules engines that were known in the art, including the RETE, TREAT, and LEAPS algorithms. *See, e.g.*, Ex. 1001, 26:45-61 (citing various prior art references). Khandekar teaches the use of known rules engines, and Sidebottom teaches the RETE and LEAP algorithms.

255. Thus, for the reasons explained above, the combination of Khandekar, Le, and Sidebottom renders this Claim obvious. *See* §VI.B.1.

C. Ground 4: Claim 17 is Rendered Obvious by Khandekar in View of Le and Stone

256. It is my opinion that a POSITA would have found Claim 17 obvious based on the combined teachings of Khandekar, Le and Stone.

1. Motivation to Combine Khandekar, Le, and Stone

257. A POSITA would have been motivated to combine Khandekar, Le, and Stone. The combination of Khandekar and Le was explained for Ground 2, and a POSITA would have been further motivated to apply Stone's teachings regarding a multi-tier web service and monitoring service-level-objectives (SLO), including its teachings regarding the monitoring of data based on those tiers. *See, e.g.*, Ex. 1007 ¶¶[0028]-[0031], [0034], [0040].

258. The combination of Khandekar and Le teaches or suggests a repository for VM and VMM images with a heuristics/rules engine that automatically deploys those images to host computers based on the monitored status of the hosts. *See*, *e.g.*, Ex. 1004, Abstract, 4:30-35, 4:58-65, 10:27-43, 13:17-35; Ex. 1005, 59:29-36, 68:21-24; *supra* §VI.A.2 (explaining how and why a POSITA would have combined Khandekar and Le's teachings), §VI.A.3[1f]-[1g], §VI.A.8.

259. Virtual machines were used in a variety of web and enterprise applications. The combination of Khandekar and Le teaches techniques for virtual machine systems, but it is not limited to any particular application. As an example, Khandekar explains that its teachings may be applied to a "web application" that includes a web server, an application server, and a database server on a network of host computers. *See* Ex. 1004 23:1-21. This three-tiered architecture was common

for web sites. Therefore, a POSITA would have found it obvious and been motivated to combine the teachings of Khandekar and Le with known web applications.

260. Like Khandekar, Stone teaches a common three-tiered architecture for web sites, with servers organized into "web server," "application," and "database" tiers. *See, e.g.*, Ex. 1007, FIG. 2:

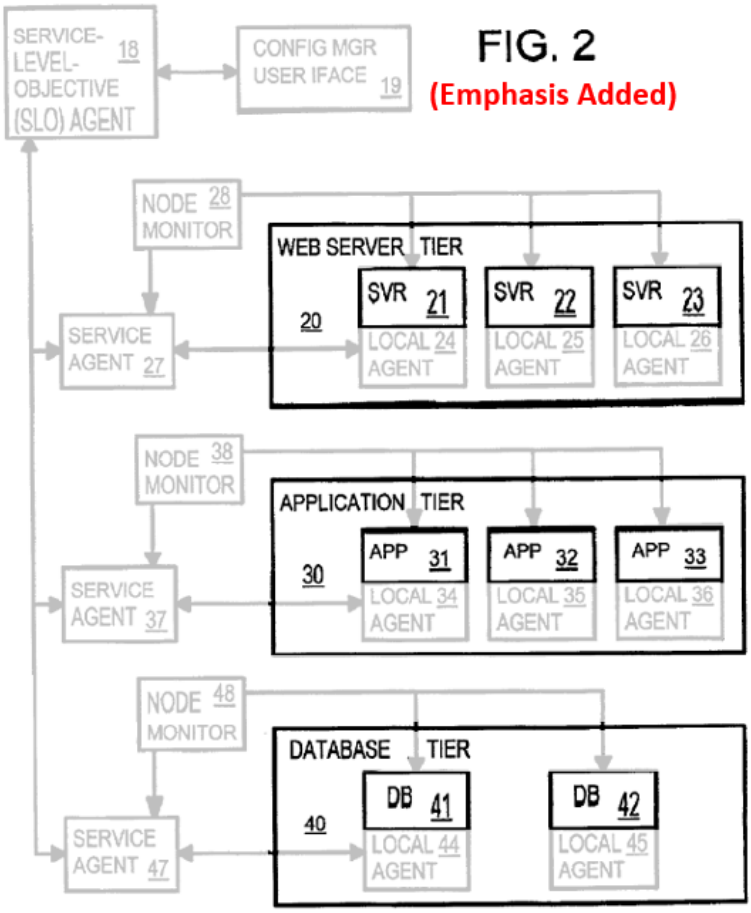


FIG. 2
(Emphasis Added)

261. A POSITA would have been motivated to combine Stone's tiered architecture with Khandekar and Le because Khandekar provides an express

motivation to apply its teachings to known web applications, including applications with Stone's three-tiered architecture, and because VMs were commonly used for web applications. Moreover, a POSITA would have been motivated to combine Stone's teachings because they offer several benefits. For example, the tiered service model allows a system administrator to enumerate the many servers, applications, databases, security devices, and other components, which are organized into functional levels or tiers that correspond to the data flow for each service. *See* Ex. 1007 ¶[0037]. Statistics collected and events monitored are thus organized by service and tier, allowing an automated management server to diagnose problems and relate them to service-level objectives. *See id.* A POSITA would have recognized that the tiered model is especially useful for diagnosing availability problems or performance bottlenecks so they can be corrected by automated management processes. *See* Ex. 1007 ¶[0038]. The tiered model also allows the system to re-adjust priorities or allocate services and tiers among the available nodes to optimize the web service automatically. *See* Ex. 1007 ¶[0055]. Thus, a POSITA would have been motivated to apply Stone's tier-based teachings to improve the distributed computing system taught by the combination of Khandekar and Le.

262. Additionally, Khandekar teaches that "the status of the hosts is monitored and the host on which a configured VM is to be deployed is selected

heuristically." Ex. 1004, 4:58-65. The deployed VMs can be monitored regularly so that an event, like the crash of the operating system in a VM, or if the load of a host crosses a threshold, can cause actions to be taken. *See* Ex. 1004, 13:44-56.

263. Stone also teaches monitoring for nodes, applying its teachings to its tiered web site architecture. When combining Stone with Khandekar and Le, a POSITA would have found it obvious and been motivated to apply the tiered architecture taught by Stone, and to apply Stone's teachings regarding the monitoring of nodes based on that tiered architecture. For example, Stone "relates to distributed computer systems, and more particularly to performance monitoring and control of server computers and applications." Ex. 1007 ¶[0001]. Stone teaches an service-level objective ("SLO") monitoring and control system that collect statistics and monitors events by tier for a multi-tier web site with a web server tier, an application server tier, and a database server tier. *See* Ex. 1007 ¶¶[0037], [0040], FIG. 2. The monitoring allows an SLO agent to take actions if an SLO is not met, such as replicating more nodes or increasing resources. *See* Ex. 1007 ¶[0057], Abstract.

264. A POSITA would have found it obvious that Stone's teachings regarding the monitoring of nodes based on tiers was advantageous for its tiered architecture. Furthermore, a POSITA would have been motivated to apply those teachings because they complement Khandekar's teachings for monitoring the

status of hosts. For example, Khandekar teaches monitoring the status, such as the load, of hosts so that actions can be taken in response. *See* Ex. 1004, 13:44-56. Stone teaches monitoring the nodes, including by service and tier, allowing an automated management process to diagnose and correct problems such as availability and performance bottlenecks. *See* Ex. 1007 ¶¶[0037]-[0038]. Automation is a central purpose behind Khandekar (*see* Ex. 1004, 3:63-4:9, 4:30-41), and thus a POSITA would have been motivated to apply Stone's tier-based monitoring teachings to improve performance and enhance automation. Stone and Khandekar teach similar monitoring of the hosts/nodes in a distributed computing system for web applications, and thus a POSITA would have been motivated to apply Stone's tier-based monitoring teachings for a multi-tier web service to Khandekar's teachings for provisioning a network of computers for a web application, as explained above. *See* Ex. 1007 ¶¶[0037], [0040]; Ex. 1004, 23:1-21.

265. A POSITA would have had a reasonable expectation of success in combining Khandekar, Le, and Stone given the similarities in their teachings, which are directed to complimentary aspects of distributed computing systems. Khandekar provides express motivation to apply its teachings to Web applications, including a three-tiered architecture, and likewise, Stone teaches a three-tiered architecture for Web sites. VMware systems were commonly used for Web applications, and the use of VMware patents Khandekar and Le for a common

Web site architecture, such as the one taught by Stone, would have been well within the intended scope of their teachings. Each of the respective teachings of the references are used in this combination for their known purpose, in the conventional manner in which they were known in the art and taught in the references. Distributed computing systems, virtual machines and related monitoring techniques were well known at the time. Given a POSITA's knowledge of distributed computing systems, virtual machines, web services, and related monitoring techniques the combination would have been well within the level of ordinary skill in the art. *See supra* §IV.

266. The combination applies known techniques (e.g., a three-tiered architecture and related monitoring techniques) to improve similar devices (e.g., web applications on a distributed computing system) in the same way as they were used in the prior art. The challenged claims are simply an obvious combination of well-known and widely practiced prior art elements according to known methods to yield predictable results.

2. Dependent Claim 17

267. It is my opinion that the combination of Khandekar, Le, and Stone renders the limitations of Claim 17 obvious.

17. The distributed computing system of claim 1, wherein the application nodes are organized into tiers, and wherein status data is collected based on the tiers.

268. The combination of Khandekar, Le, and Stone teaches or suggests the distributed computing system of claim 1 (*see supra* §VI.A.3), wherein the application nodes are organized into tiers, and wherein status data is collected based on the tiers.

269. Khandekar and Le both teach monitoring the status of host computers, For example, Khandekar teaches collecting status data of the hosts. *See, e.g.*, Ex. 1004, 4:58-65 ("… the status of the hosts is monitored and the host on which a configured VM is to be deployed is selected heuristically."), 12:57-13:8 ("Information about the current state of the different hosts is preferably input dynamically into a database or other data structure in the heuristics engine 880."), 13:44-56 ("The deployed VMs can be monitored regularly so that any unexpected event, like the crash of the operating system in the VM, can be reported to the user or administrator. … Hosts can be monitored for load and if the load crosses a threshold, the administrator can be alerted and/or actions can be taken …"), 23:28-51 ("… the status of a plurality of hosts platforms is monitored and the appropriate staged VM is determined and selected from the set of staged VMs."). For example, Khandekar teaches monitoring the load on hosts. *See* Ex. 1004, 13:44-56 ("Hosts can be monitored for load and if the load crosses a threshold …").

270. Le also teaches status data is collected. *See* Ex. 1005, 68:21-24 ("… the agent 7300 can also report status and system performance information to the imaging server 2101, allowing the UCMS to detect the presence of the agent and to monitor the deployed computer's health.").

271. Khandekar teaches monitoring status for a web server, an application server, and a database server on a network of host computers (*see, e.g.* Ex. 1004 4:58-65, 23:1-21), while Le teaches monitoring its deployed machines (*see* Ex. 1005, 68:21-24). A POSITA would have been motivated to combine Khandekar and Le with Stone's teachings regarding the monitoring of status data in a tiered distributed computing system, including web server, application, and database tiers. *See supra* §VI.C.1 (explaining how and why a POSITA would have combined the teachings of Khandekar, Le, and Stone).

272. Stone teaches a distributed computing system. *See* Ex. 1007 ¶[0001] ("This invention relates to distributed computer systems, and more particularly to performance monitoring and control of server computers and applications."), ¶[0026] ("These services are provided by a complex set of applications that may be distributed among several different computers networked together."). In the distributed computing system, Stone teaches organizing application nodes into tiers. For example, Stone teaches "a multi-tiered service model of an e-business web service" where "[f]our levels or tiers of service components are provided":

"firewall devices 11, 12" on "firewall tier 10"; "web servers 21, 22, 23 on web server tier" 20; "[a]pplication servers 31, 32, 33 on application tier 30"; and "databases 41, 42" on "database tier 40." *See* Ex. 1007 ¶¶[0028]-[0031], [0034] ("Each service can vary in the arrangement and type of tiers, as well as the number of nodes for each tier."), FIG. 1:

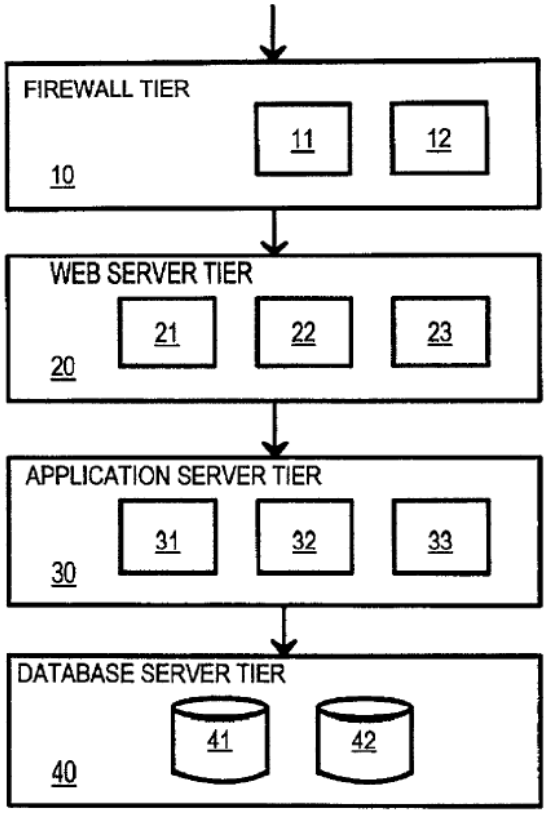


FIG. 1

273. Stone teaches status data is collected based on the tiers, including "web server," "application," and "database" tiers. For example, Stone teaches "a service-level-objective (SLO) monitoring and control system for a multi-tier web

site." *See, e.g.*, Ex. 1007 ¶[0040], [0037] ("The tiered model organizes these components into functional levels or tiers that correspond to the data flow for each Service. ***Statistics collected and events monitored can thus be organized by Service and tier***, allowing an automated management Server to diagnose problems and relate them to Service-level objectives.") (emphasis added), FIG.2:

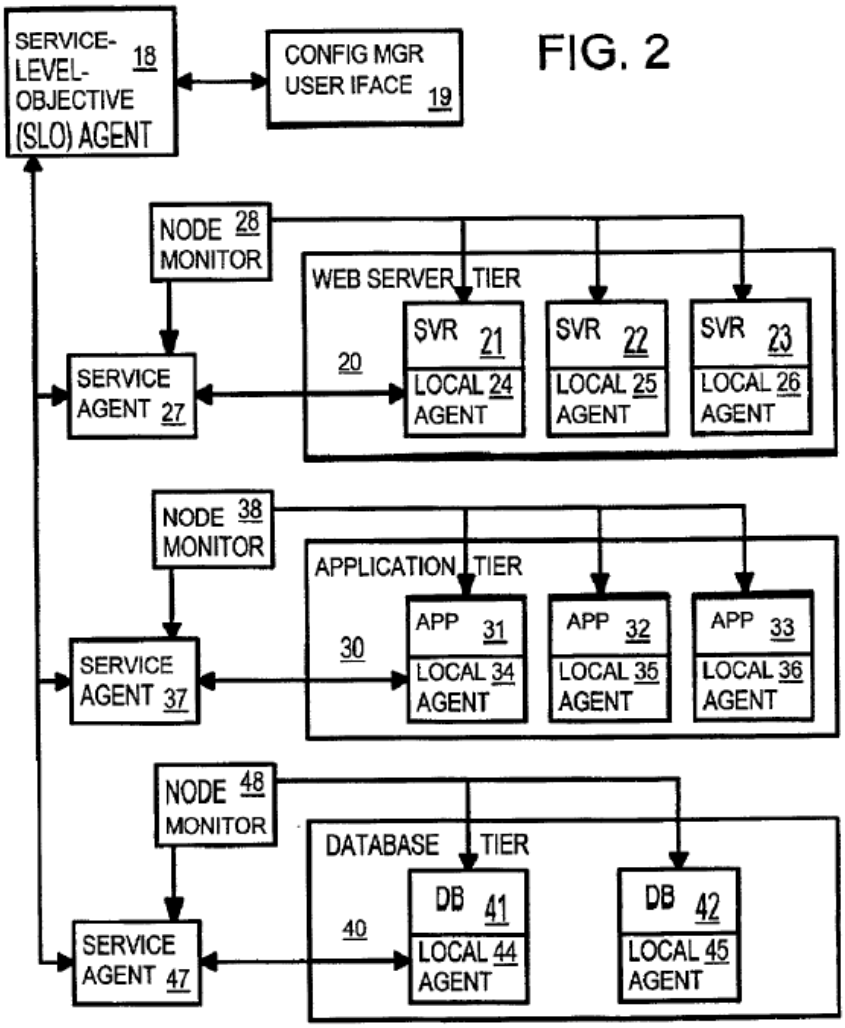


FIG. 2

274. Stone teaches "service agents" that collect status data based on the tiers, for example, service agent 27 collect status data for the web server tier 20. *See* Ex. 1007 ¶¶[0042]-[0044]:

> Several mid-level agents are used in this embodiment. SLO agent 18 forwards service information to service agents 27, 37, 47. Although the overall service-level objectives are kept by SLO agent 18, service agents 27, 37, 47 keep track of service information and status for service components running on multiple systems controlled by these service agents 27, 37, 47.
>
> The lower-level local agents are configured by service agents 27, 37, 47. In this embodiment each computer server machine has its own local agent running on it. Each local agent keeps information about service components running on its system. Each local agent monitors dependencies (other software modules) required by its services, and general system and service performance information, such as CPU, memory, and I/O channel usage.
>
> Service agent 27 controls local agents 24, 25, 26, which control web servers 21, 22, 23, respectively on web-server tier 20.

Because agent 27 controls the local agents that monitor service performance information for the web-server tier 20, a POSITA would have found it obvious that service agent 27 collect status data for web server tier 20. *See id.* Similarly, it would have been obvious that service agents 37 and 47 collect status data for the application tier 30 and database tiers, respectively. *See* Ex. 1007 ¶¶[0042]-[0046].

275. Stone also teaches "node monitors" that collect status data based on the tiers, for example, node monitor 28 which collects status data for the web server tier 20. *See* Ex. 1007 ¶[0044]:

> Node monitor 28 monitors the nodes for web servers 21, 22, 23, and can also monitor any sub-networks associated with these nodes. When an entire node fails, service agent 27 is notified by node monitor 28 even when the local agent is unable to communicate with service agent 27 due to the failed network connection or crashed server or hardware.
>
> For example, when the node containing web server 22 fails, local agent 25 running on the same machine is unable to notify service agent 27. Instead, node monitor 28 notifies Service agent 27 of the node's failure.

Similarly, node monitors 38 and 48, collect status data for the application tier 30 and database tier, respectively. *See* Ex. 1007 ¶[0046].

276. In view of Stone's teachings, explained above, a POSITA would have found it obvious to organize application nodes, such as Khandekar's hosts, into tiers, including web server, application, and database tiers. Indeed, Khandekar already teaches or suggests organizing hosts into those tiers (*see, e.g.* Ex. 1004 4:58-65, 23:1-21), and the obviousness of doing so are further reinforced by Stone's teachings. Additionally, Stone teaches collecting status data based on tiers, including web server, application, and database tiers. A POSITA would have found

it obvious for status data to be collected based on those tiers because all three references teach monitoring node/host status, and Stone teaches an obvious way to organize the collection of status data when application nodes are organized into tiers, as explained above. Stone teaches an improved way to monitor a web service in a tiered environment. *See, e.g.*, Ex. 1007 ¶[0038]. *See supra* §VI.C.1. Therefore, the combination renders claim 17 obvious.

277. This is further confirmed by the '138 patent, which similarly discloses that application nodes are organized into the same three tiers as Stone (web server, application, and database tiers), and status data is collected based on the tiers. *See, e.g.*, Ex. 1001, 13:40-56 ("Tier level 31C includes one or more tiers within each domain that represent the functional subcomponents of applications and services provided by the domain. For example, storefront domain 34A includes **a web server tier** (labeled 'web tier') 36A, a **business application tier** (labeled 'app tier') 36B, and a **database tier** (labeled 'DB tier') 36C. Web server tier 36A, business application tier 36B and database tier 36C interact with one another to present a customer with an online storefront application and services. …") (emphasis added), Fig. 2:

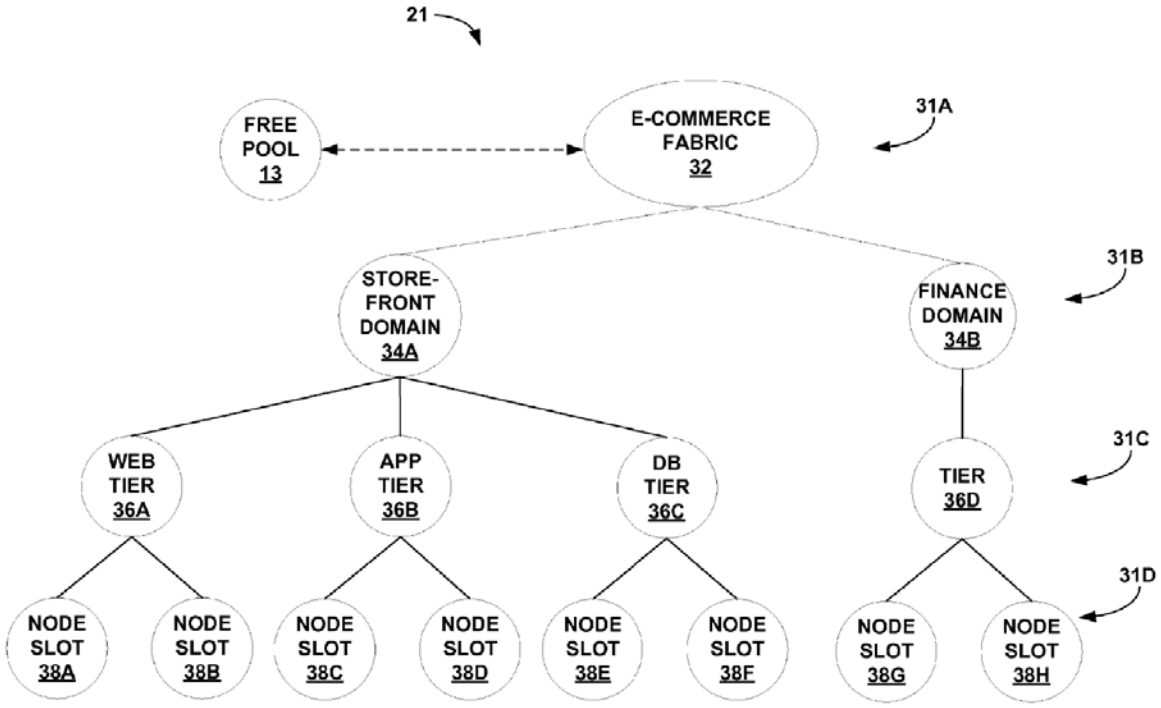


FIG. 2

278. Thus, for the reasons explained above, the combination of Khandekar, Le, and Stone renders this limitation obvious. *See* §VI.C.1.

VII. BACKGROUND AND QUALIFICATIONS

279. This section contains a summary of my educational background, career history, publications, and other relevant qualifications. My full curriculum vitae is attached as Appendix 1 to this declaration.

280. I am a Distinguished Professor of Computer Science and Director of the Center for Smart and Connected Society at the University of Massachusetts. I have more than two decades of experience in distributed systems, operating systems, data management, and networking.

281. I received my Bachelor of Technology (B.Tech.) degree in 1993 from the Department of Computer Science and Engineering at the Indian Institute of Technology (IIT) Mumbai, India. I was awarded the Institute Medal for being the top ranking undergraduate student in Computer Science and Engineering by IIT Bombay. In 1994, I received my Master of Science (M.S.) degree in Computer Sciences from The University of Texas at Austin. In 1998, I received my Doctor of Philosophy (Ph.D.) also from the Department of Computer Sciences at The University of Texas at Austin. My dissertation, which was in the area of file systems, was recognized by the department as the Best Doctoral Dissertation of 1998-1999.

282. From 1998 until 2009, I worked as an Assistant and then Associate Professor at the Department of Computer Science at the University of Massachusetts Amherst. Since 2009, I have been working as a Professor at the College of Information and Computer Sciences at the University of Massachusetts Amherst. In 2020, I was appointed as a Distinguished Professor of Computer Science by the university. In addition, I also serve as Associate Dean of the College of Information and Computer Sciences and as the Director of the Center for Smart and Connected Society within the college. I currently work in all three capacities.

283. I am a Fellow of the ACM, the IEEE and the AAAS. My research has won many awards, including a Test of Time award for my research paper on multi-tier web applications.

284. As a faculty member at the University of Massachusetts, I have taught graduate courses on distributed and operating systems and undergraduate courses on operating systems. The courses covered in-depth discussions of distributed systems and applications and operating systems concepts such as virtualization, file systems, process and memory management.

285. I have published over 270 technical papers that have been cited in other publications more than 20,000 times. I have also published three book chapters and am a co-inventor on four US patents in operating and distributed systems. Over the years, I have published papers on topics such as virtualization, dynamic application provisioning, data management, web applications, and distributed systems.

A. Compensation

286. For my efforts in connection with the preparation of this declaration I have been compensated at my standard rate for this type of consulting activity. My compensation is in no way contingent on the results of these or any other proceedings relating to the above-captioned patent.

B. Materials and Other Information Considered

287. I have considered information from various sources in forming my opinions. I have reviewed and considered each of the exhibits listed in the attached Appendix 2 (Materials Considered in the Preparation of This Declaration) in forming my opinions.

VIII. UNDERSTANDING OF THE LAW

288. I am not an attorney. In forming my opinions in this Declaration, I applied the relevant legal principles provided to me by counsel, which are summarized in Appendix 4.

IX. RESERVATION OF RIGHTS

289. My opinions are based upon the information that I have considered to date. I am unaware of any evidence of secondary considerations with respect to the '138 Patent that would render any of the challenged claims non-obvious. I reserve the right, however, to supplement my opinions in the future to respond to any arguments raised by the owner of the '138 Patent and to take into account new information that becomes available to me.

290. I declare that all statements made herein of my knowledge are true, and that all statements made on information and belief are believed to be true, and 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 United States Code.

Executed on September 09, 2020.

By:

Prashant Shenoy

APPENDIX 1: CURRICULUM VITAE OF PRASHANT SHENOY

Prashant Shenoy

College of Information and Computer Sciences
140 Governor's Drive
University of Massachusetts
Amherst, MA 01003-4610

URL: http://www.cs.umass.edu/~shenoy
E-mail : shenoy@cs.umass.edu
Phone: (413) 577-0850
Fax: (413) 545-1249

Research Interests

Operating and Distributed systems, Sensor networks, Mobile and Multimedia systems, Sustainability

Education

August 1998	Doctor of Philosophy (Ph.D.), Department of Computer Sciences, The University of Texas of Austin. Dissertation title: *Symphony: An Integrated Multimedia File System* Advisor: *Prof. Harrick M. Vin*.
December 1994	Master of Science (M.S) in Computer Sciences The University of Texas of Austin.
July 1993	Bachelor of Technology (B.Tech), Department of Computer Science and Engineering The Indian Institute of Technology, Bombay, India.

Work Experience

- 7/2020-present: Distinguished Professor, College of Information and Computer Sciences, University of Massachusetts Amherst
- 9/2017 - present, Director, Center for Smart and Connected Society
- 11/2018 - 8/2019, Director, IALS Center for Personal Health Monitoring
- 9/2016 - present: Associate Dean, College of Information and Computer Sciences, University of Massachusetts Amherst (part-time appointment)
- 9/2009-present: Professor, College of Information and Computer Sciences, University of Massachusetts Amherst
- 2017-2018: Visiting Faculty, IIT Bombay, India.
- Fall 2011: Visiting Researcher, NICTA, Sydney, Australia
- 9/2004-8/2009: Associate Professor, Department of Computer Science, University of Massachusetts Amherst.
- 9/1998-8/2004: Assistant Professor, Department of Computer Science, University of Massachusetts Amherst.
- 8/1995-8/1998: Graduate Research Assistant, Department of Computer Sciences, University of Texas at Austin.
- 5/1995-8/1995: Summer Intern, AT&T Bell Laboratories, Murray Hill, NJ.
- 8/1994-5/1995: Teaching Assistant, Department of Computer Sciences, University of Texas at Austin.

- 5/1994-8/1994: Summer Intern, Microsoft Research, Redmond, WA.

Expert Witness Experience

- Veeam v. Symantec, *Inter Partes Review* Case IPR2013-00141 for US Patent 6,931,558, "Computer Restoration Systems and Methods" Deposition October 2013 and March 2014.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2013-00142 for US Patent 6,931,558, "Computer Restoration Systems and Methods," Deposition October 2013 and March 2014.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2013-00143 for US Patent 7,191,299, "Method and System of Providing Periodic Replication," Deposition November 2013 and March 2014.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2013-00144 for US Patent 7,254,682, "Selective File and Folder Snapshot Image Creation," Written testimony February 2013.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2013-00145 for US Patent 7,254,682, "Selective File and Folder Snapshot Image Creation," Written testimony February 2013.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2013-00150 for US Patent 7,093,086, "Disaster Recover and Backup Using Virtual Machines," Deposed November 2013 and March 2014.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2014-00088 for US Patent 7,480,822, "Recover and Operation of Captured Running States from Multiple Computing Systems On a Single Computing System," Deposition June 2014.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2014-00089 for US Patent 7,831,861, "Techniques for Efficient Restoration of Granular Application Data" Deposition June 2014.
- Veeam v. Symantec, *Inter Partes Review* Case IPR2014-00091 for US Patent 8,117,168, "Methods and Systems for Creating and Managing Backups Using Virtual Disks," Deposition July 2014.
- SAP and Riverbed v. Parallel Networks, *Inter Partes Review*, Case IPR2014-01398 for US Patent 8,352,570 "Method and System for Uniform System Locator Transformation," Written testimony August 2014.
- SAP and Riverbed v. Parallel Networks, *Inter Partes Review*, Case IPR2014-01399 for US Patent 7,571,217 "Method and System for Uniform System Locator Transformation," Written testimony August 2014.
- Delphix v. Actifio, *Inter Partes Review* Case IPR2015-00100 for US Patent 8,566,361, "Datacenter Workflow Automation Scenarios Using Virtual Databases," August 2015. Deposed.
- Delphix v. Actifio, *Inter Partes Review Case* IPR2015-00108 for US Patent 8,566,361, "Datacenter Workflow Automation Scenarios Using Virtual Databases," August 2015. Deposed.
- Delphix v. Actifio, *Inter Partes Review Case* IPR2015-01678 for US Patent 8,299,944, "System and Method for Creating Deduplicated Copies of Data Storing Non-lossy Encodings of Data Directly in a Content Addressable Store," August 2015. Deposed.
- Delphix v. Actifio, *Inter Partes Review* Case IPR2015-01689 for US Patent 8,788,769, "System and method for performing backup or restore operations utilizing difference information and timeline state information," August 2015. Deposed.

- Cisco v. Egenera, *Inter Partes Review* Case IPR2017-01341 for U.S. Patent No. 7,231,430, "Reconfigurable Virtual Processing System, Cluster, Network and Method," Deposed, January 2018.
- Cisco v. Egenera, *Inter Partes Review* Case IPR2017-01340 for U.S. Patent No. 6,971,044, "Service Clusters and Method in a Processing System with Failover Capability," Deposed January 2018.
- Cisco v. HP Enterprise, *Inter Partes Review* Case IPR2017-01933 for U.S. Patent No. 8,478,799, "Namepsace File System Accessing An Object Store," Written testimony, August 2017
- EMC, Lenovo and NetApp v. IV, *Inter Partes Review* Case IPR2017-00374 for U.S. Patent no 8,275,827, "Software-based Network Attached Storage Services Hosted On Massively Distributed Parallel Computing Networks." January 2018, Deposed.
- EMC, Lenovo and NetApp v. IV, *Inter Partes Review* Case IPR2017-00439 for U.S. Patent no 8,275,827, "Software-based Network Attached Storage Services Hosted On Massively Distributed Parallel Computing Networks." January 2018, Deposed.
- Lenovo and EMC v. IV, *Inter Partes Review* Case IPR2017-00477 for U.S. Patent no 8,387,132, " Information Packet Communication With Virtual Objects" February 2018, Deposed.
- Amazon and Amazon Web Services v. Personal Web Technologies and Level 3 Communications, *US District Court Patent Litigation*, Written testimony for US Patent nos 6,928,442; 7,802,310; 7,945,544; and 8,099,420. December 2018.
- Hybir v. Veeam, *Inter Partes Review* for U.S. Patent no 8,051,043, "Group Based Complete and Incremental Computer Field Backup System, Process and Apparatus," Written testimony, May 2020.
- Hybir v. Veeam, *Inter Partes Review* for U.S. Patent no 9,679,146, "Group Based Complete and Incremental Computer Field Backup System, Process and Apparatus," Written testimony, May 2020.
- Hybir v. Veeam, *Inter Partes Review* for U.S. Patent no 9,037,545, "Group Based Complete and Incremental Computer Field Backup System, Process and Apparatus," Written testimony, May 2020.

Highlights of Research Accomplishments

- Seventeen publications have received best paper or best paper runner-up awards. One paper received ACM Sigmetrics Test of Time award.
- Have published over 270 technical papers that have been cited more than 20,000 times. H-index of my publications is 74 as per Google Scholar.
- My research has received more than $35M in federal funding from NSF, DoD, NASA, EPA and state funding from MA DOER.
- Technical area lead and UMass PI for Internet of Battlefield Things (IoBT) center funded by Army Research Labs, Oct 2017.
- Principal Investigator on a MA DOER award to establish the UMass Clean Energy center to conduct applied research and outreach in the area of clean energy. Have worked with sate policy makers on clean energy issues as part of this effort.
- Awarded NSF Partnerships for Innovation (PFI) grant to conduct translational research and pilot deployments of energy efficiency technologies developed by my group.
- Collaboration with Holyoke Gas & Electric on big data and energy topics using smart meters.
- Have closely collaborated with the MGHPCC since its inception. Secured a multi-university NSF MRI award and an award from Mass. Life Sciences Center (MLSC) to deploy two large compute clusters for HPC and data science research at the MGHPCC.
- Have established several large research testbeds for the research community, including two GENI networking testbeds (DiCloud and ViSE), Smart* testbeds and the MassNZ solar powered data center.
- Served on the steering committees of IEEE Transactions on Mobile Computing, ACM BuildSys and ACM Sigcomm e-Energy. Current Chair of ACM e-Energy Steering committee.
- Received UMass President's Science and Technology Award thrice (2017, 2012 and 2007).
- BenchLab performance benchmarking software developed by my group has been downloaded by more than 9000 researchers and practitioners from 77 countries.

Honors and Awards

- Fellow of the ACM, 2019.
- Fellow of the AAAS, 2018.
- Fulbright Specialist Scholar, 2017-2018
- ACM Sigmetrics 2016 Test of Time Award.
- Fellow of the IEEE, 2013
- Distinguished member of the ACM, 2009.
- Best Paper Awards or Finalists:

- Best paper runner-up award, *ACM e-Energy* 2020 Conference
- Best paper award, *ACM e-Energy* 2019 Conference
- Best paper finalist, *ACM HPDC* 2019 Conference
- Best paper finalist, *ACM Buildsys* 2017.
- Best paper Finalist, *ACM Buildsys* 2014.
- Best paper, *ACM/IEEE IWQOS* symposium, Montreal, 2013
- Best paper runner-up, *ACM eEnergy* conference, Berkeley, CA, 2013
- Best papers of *IEEE PerCom 2012* conference (one of three papers chosen for this honor).
- Best paper, *IEEE COMSNETS 2012* conference, India.
- Best paper, *ACM Sigcomm GreenNets 2011* workshop, Toronto, Canada.
- Best papers of ACM VEE 2009 conference (one of four papers chosen for this honor)
- Best paper, *Usenix 2007* annual technical conference.
- Best Paper, *ACM Multimedia 2005*, Singapore.
- Best Student Paper Finalist, *ACM Multimedia 2005*.
- Best Student Paper award, *IEEE Autonomic Computing conference (ICAC) 2005.*
- Best Paper, *IEEE Web Information Systems Engineering Conference*, 2002.
- Best Paper in Performance/Systems Category *World Wide Web Conference*, May 2002.

- Keynote Talks and Distinguished Lecture Talks

  - Distinguished Lecturer, Univ of Maryland Baltimore Country (May 2019), Colorado State University (March 2018)
  - Keynote Speaker, Asplos Edge Computing Workshop 2019,
  - Keynote Speaker, CPSWeek Fog Computing Workshop 2019,
  - Keynote Speaker, Cloud Control 2018
  - Keynote Speaker, ACM ICPE 2013 conference
  - Keynote Speaker, ACM GreenMetrics 2013
  - Keynote Speaker, IEEE E6 Energy Workshop, 2012.
  - Keynote speaker, TCS Excellence in Computer Science (TECS) Week, January 2012.

- Other honors and awards:

  - Conti Reseaarch Fellowship, 2020-2021.
  - NASA JSC Director's Innovation Group Achievement Award, Oct 2017 (given to NASA REALM project that we are part of)
  - NASA AES Innovation Award 2017 (given to NASA REALM project that we are part of)
  - UMass President Science and Technology Award, 2017, 2012 and 2007
  - Google Faculty Award, 2016 and 2013
  - IBM Faculty Partnership Award, June 2000, June 2001 and June 2003.

- AT&T VURI Award
- Lilly Foundation Teaching Fellowship, 2001-2002
- National Science Foundation (NSF) CAREER Award, April 2000.
- Best Doctoral Dissertation of 1998-99, Department of Computer Sciences, University of Texas at Austin.
- MCD Fellow, University of Texas at Austin, 1993-95.
- Institute Medal recipient for being the top ranking graduating student, Department of Computer Science and Engineering, Indian Institute of Technology, Bombay, July, 1993.
- JN Tata Scholarship, 1993.

Books and Book Chapters

B1. K. Ramamritham, P. Shenoy, and G. Karmarkar, "Energy Informatics: A Computational Perspective," Book In preparation, Scheduled for publication in 2021.

B2. Michael Zink and Prashant Shenoy, "Caching and Distribution Issues for Streaming Content Distribution Networks," X. Tang, J. Xu, and S T. Chanson (eds), Springer, 2005.

B3. Harrick Vin and Prashant Shenoy, "Storage Architectures for Digital Imagery," *Image Databases: Search and Retrieval of Digital Imagery*, V. Castelli and L. Bergman (eds), John Wiley, 2002.

B4. Prashant Shenoy and Harrick Vin, "Media Servers," *Readings in Multimedia Computing*, K Jeffay and H. Zhang (eds), Morgan Kaufman Publishers, August 2001.

Journal Publications

J1. Saurabh Bagchi, Tarek F. Abdelzaher, Ramesh Govindan, Prashant Shenoy, Akanksha Atrey, Pradipta Ghosh, and Ran Xu, "New Frontiers in IoT: Networking, Systems, Reliability, and Security Challenges," IEEE IoT Journal 2020.

J2. Srini Iyengar, Stephen Lee, David Irwin, Prashant Shenoy, and Ben Weil, "WattScale: A Data-driven Approach for Energy Efficiency Analytics of Buildings at Scale," ACM Transactions on Data Science, 2020.

J3. Muhammad Aftab, Sid Chi-Kin Chau and Prashant Shenoy, "Efficient Online Classification and Tracking on Resource-constrained IoT Devices," ACM Transactions on Intenet of Things (ACM TIOT), 2020.

J4. Khaled Zaouk, Fei Song, Chenghao Lyu, Arnab Sinha, Yanlei Diao, Prashant Shenoy, "UDAO: A Next-Generation Unified Data Analytics Optimizer," PVLDB, vol 12, no 12, pp 1934-1937, 2019.

J5. Sean Barker, Sandeep Kalra, David Irwin, and Prashant Shenoy, "Building Virtual Power Meters for Online Load Tracking," ACM Transactions on Cyber-Physical Systems (ACM TCPS), vol 3, no 2, March 2019.

J6. T. Abdelazer, P. Shenoy, et. al. "Towards an Internet of Battlefield Things: A Resilience Perspective," IEEE Computer, Special Issue of Cyber-physiscal Systems Resilience, 2018.

J7. N. Bashir, D. Irwin, P. Shenoy, and J. Taneja, "Mechanisms and Policies for Controlling Distributed Solar Capacity," ACM Transactions on Sensor Networks (ACM TOSN), 2018

J8. Srinivasan Iyengar, Sandeep Kalra, Anushree Ghosh,David Irwin, Prashant Shenoy, and Benjamin Marlin "Inferring Smart Schedules for Dumb Thermostats," *ACM Transactions on Cyber-Physical Systems (ACM TCPS)*, 2018

J9. Tian Guo and Prashant Shenoy, "Providing Geo-Elasticity in Geographically Distributed Clouds," *ACM Transactions on Internet Technology (TOIT)*, 2018

J10. Tian Guo and Prashant Shenoy, "Performance and Cost Considerations for Providing Geo-Elasticity in Database Clouds," *ACM Transactions on Adaptive and Autonomous System (TAAS)*, 2018

J11. Tian Guo, Prashant Shenoy, K. K. Ramakrishnan, and Vijay Gopalakrishnan, "Latency-aware Virtual Desktops Optimization in Distributed Clouds," *Multimedia Systems Journal (MMSJ)*, 2018

J12. P. Sharma, S. Lee, T. Guo, D. Irwin and P. Shenoy, "Managing Risk in a Derivative IaaS Cloud,", *IEEE Trans on Parallel and Distributed Systems (TPDS)*, 2017.

J13. Sean Barker, David Irwin, and Prashant Shenoy, "Pervasive Energy Monitoring and Control through Low-Bandwidth Power Line Communication," *IEEE Internet of Things Journal*, 2017.

J14. David Irwin, Srinivasan Iyengar, Stephen Lee, Aditya Mishra, Prashant Shenoy, Ye Xu, "Enabling Distributed Energy Storage by Incentivizing Small Load Shifts," *ACM Transactions on Cyber-Physical Systems (ACM TCPS)*, 2017.

J15. Srinivasan Iyengar, Navin Sharma, David Irwin, Prashant Shenoy, Krithi Ramamritham, "A Cloud-based Black Box Solar Predictor for Smart Homes," *ACM Transactions on Cyber- Physical Systems - Special Issue on Smart Homes, Buildings, and Infrastructures*, 2017.

J16. Zhichuan Huang, Ting Zhu, Yu Gu, David Irwin, Aditya Mishra, Daniel Menasche, Prashant Shenoy, "Minimizing Transmission Loss in Smart Microgrids by Sharing Renewable Energy," *ACM Trans on Cyber-Physical Systems*, vol 1, number 2, 2017.

J17. Prateek Sharma, Patrick Pegus, David Irwin, and Prashant Shenoy, "Design and Operational Analysis of a Green Data Center," *IEEE Internet Computing*, Special Issue on Energy-efficient Data Centers, vol 21, no 4, July 2017.

J18. Prateek Sharma, David Irwin, and Prashant Shenoy, "Keep it Simple: Bidding for Servers in Today?s Cloud Platforms," *IEEE Internet Computing*, vol 21, no 3, May 2017.

J19. Navin Sharma, Dilip Krishnappa, Sean Barker, David Irwin, and Prashant Shenoy, "Managing server clusters on intermittent power," *PeerJ Computer Science Journal*, December 2015.

J20. Boduo Li, Yanlei Diao, Prashant Shenoy, "Supporting Scalable Analytics with Latency Constraints," *PVLDB Journal*, 8(11), 2015. Also presented at the VLDB conference, Hawaii, Aug 2015.

J21. Dong Chen, Sandeep Kalra, David Irwin, Prashant Shenoy, and Jeannie Albrecht, "Preventing Occupancy Detection from Smart Meter Data," *IEEE Transactions on Smart Grid*, 2015.

J22. T. Wood, K K. Ramakrishnan, P. Shenoy, J. van der Merwe, J. Hwang, G. Liu and L. Chaufournier, "CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines," *IEEE/ACM Transactions on Networking (TON)*, 2014.

J23. Navin Sharma, Jeremy Gummeson, David Irwin, Ting Zhu and Prashant Shenoy "Leveraging weather forecasts in renewable energy systems," *Sustainable Computing: Informatics and Systems*, Volume 4, Issue 3, pages 160-171, September 2014.

J24. Vimal Mathew, Ramesh K. Sitaraman and Prashant Shenoy "Energy-efficient content delivery networks using cluster shutdown," *Sustainable Computing: Informatics and Systems (SUSCOM)*, Special Issue of Invited Papers from IGCC 2013, 2014.

J25. Navin Sharma, David Irwin, and Prashant Shenoy, "BlinkFS: A Distributed File System for Intermittent Power," *Sustainable Computing (SUSCOM)*, 2014.

J26. Rahul Singh, Prateek Sharma, David Irwin, Prashant Shenoy, and K.K. Ramakrishnan, "Here Today, Gone Tomorrow: Exploiting Transient Servers in Data Centers," *IEEE Internet Computing*, 18(4):22-29 2014.

J27. Sean Barker, Sandeep Kalra, David Irwin, and Prashant Shenoy, "Empirical Characterization, Modeling, and Analysis of Smart Meter Data," *IEEE Journal on Selected Areas in Communications (JSAC)*, Special Series on Smart Grid Communications, 32(7):1312-1327, July 2014.

J28. Tian Guo, Upendra Sharma, Prashant Shenoy, Timothy Wood, Sambit Sahu, "Cost-aware Cloud Bursting for Enterprise Applications," *ACM Transactions on Internet Technology (TOIT)*, vol 13, no 3, May 2014.

J29. Prashant Shenoy, "Multimedia Systems Research: The First Twenty Years and Lessons for the Next Twenty," *ACM Transactions on Multimedia Computing, Communications and Applications (ACM TOMCCAP),* Special Issue on 20 Years of ACM Multimedia, vol 9, no 1s, October 2013.

J30. R. Singh, P. Shenoy, M. Natu, V. Sadaphal and H. Vin, "Analytical Modeling for What-if Analysis in Complex Cloud Computing Applications," *ACM Sigmetrics Performance Evaluation Review, Special Issue on Cloud Computing*, Vol 40, no 4, pages 53-62, March 2013

J31. A. Mishra, D. Irwin, P. Shenoy, J. Kurose, T. Zhu, "GreenCharge: Managing Renewable Energy in Smart Buildings," *IEEE Journal on Selected Areas in Communications (JSAC), Special Series on Smart Grid Communications*, 31(7): 1281-1293, July 2013.

J32. B. Li, E. Mazur, Y. Diao, A. McGregor and P. Shenoy, "SCALLA: A Platform for Scalable One-Pass Analytics using MapReduce," *ACM Transactions on Database Systems (TODS)*, 37:4, December 2012. Special Issue of Best Papers of Sigmod 2011.

J33. Peter Desnoyers, Timothy Wood, Prashant Shenoy, Rahul Singh, Sangameshwar Patil, and Harrick Vin, "Modellus: Automated Modeling of Complex Internet Data Center Applications," *ACM Transactions on the Web (TWEB)*, 6(2), 1-19, May 2012.

J34. T. Wood, K K Ramakrishnan, P. Shenoy, K Van der Merwe, "Enterprise Ready Virtual Cloud Pools: Vision, Oppurtunities and Challenges," *The Computer Journal, Oxford Univ. Press*, vol 55, number 8, pages 1-10, October 2012.

J35. Navin Sharma, David Irwin, Prashant Shenoy and Michael Zink, "MultiSense: Proportional-Share for Mechanically Steerable Sensor Networks," Multimedia Systems Journal (MMSJ), vol 18, pages 425-444, October 2012.

J36. Y. Nie, R. Cocci, Z. Cao, Y. Diao, and P. Shenoy "SPIRE: Efficient Data Interpretation and Compression over RFID Streams," *IEEE Trans. on Knowledge and Data Engineering (TKDE)*, vol 24, number 1, pages 141-155, 2012.

J37. Zhao Cao, Charles Sutton, Yanlei Diao, and Prashant Shenoy,"Distributed Inference and Query Processing for RFID Tracking and Monitoring" *PVLDB Journal*, vol 4, number 5, pages 326-337, March 2011.

J38. Xiaotao Liu, Mark D. Corner, Prashant Shenoy, "Ferret: An RFID-enabled pervasive multimedia application," Ad Hoc Networks, 9(4), 2011.

J39. Jeremy Gummeson, Deepak Ganesan, Prashant Shenoy and Mark Corner, "An Adaptive Link Layer for Heterogeneous Multi-radio Mobile Sensor Networks," *IEEE Journal on Selected Areas in Communications (JSAC), Special Issue on Simple Wireless Sensor Networking Solutions*, vol 28, no 7, pages 1094-1104 Sept 2010.

J40. Gaurav Mathur, Peter Desnoyers, Paul Chukiu, Deepak Ganesan and Prashant Shenoy, "Ultra Low Power Data Storage for Sensor Networks," *ACM Transaction on Sensor Networks (TOSN)*, 2009.

J41. David Yates, Erich Nahum, James F. Kurose, Prashant Shenoy, "Data Quality and Query Cost in Pervasive Sensing Systems", *Elsevier Pervasive and Mobile Computing Journal*, Special Issue of Selected Papers from IEEE Percom'08, 2009.

J42. Ming Li, Deepak Ganesan and Prashant Shenoy, "PRESTO: Feedback-driven Data Management in Sensor Networks," *IEEE/ACM Transactions on Networking (TON)*, October 2009.

J43. Xiaotao Liu, Mark D. Corner, and Prashant Shenoy, "SEVA: Sensor Enhanced Video Annotation," *ACM Transactions on Multimedia Computing, Communications and Applications (ACM TOMCCAP)*, August 2009.

J44. Bhuvan Urgaonkar, Prashant Shenoy, and Timothy Roscoe. "Resource Overbooking and Application Profiling in a Shared Internet Hosting Platform," *ACM Transactions on Internet Technologies (TOIT)*, 2008.

J45. Xiaotao Liu, Prashant Shenoy, and Mark D. Corner, "Chameleon: Application Level Power Management," *IEEE Transactions on Mobile Computing*, 7(8):995-1010, August 2008.

J46. Bhuvan Urgaonkar and Prashant Shenoy "Cataclysm: Scalable Overload Policing for Internet Applications," *Elsevier Journal of Network and Computer Applications (JNCA)*, vol 31, pages 891-920, July 2008.

J47. B. Wang, J. Kurose, P. Shenoy and D. Towsley, "Multimedia Streaming via TCP: An Analytic Performance Study," *ACM Transactions on Multimedia Computing, Communications and Applications (TOMCCAP)*, 4(2), April 2008.

J48. A. Chandra and P. Shenoy. "Hierarchical Scheduling for Symmetric Multiprocessors," *IEEE Trans. On Parallel and Distributed Systems (TPDS)*, 19(3), pp 418-431, March 2008.

J49. Bhuvan Urgaonkar, Prashant Shenoy, Abhishek Chandra, Pawan Goyal, and Timothy Wood, "Agile Dynamic Provisioning of Multi-tier Internet Applications", *ACM Transactions on Adaptive and Autonomous Systems (TAAS)*, Vol. 3, No. 1, pages 1-39, March 2008.

J50. B. Urgaonkar, A. Rosenberg and P. Shenoy, "Application Placement on a Cluster of Servers", *International Journal of Foundations of Computer Science*, Vol. 18, No. 5, pages 1023-1041, October 2007

J51. Z. Ge, P. Ji and P. Shenoy, "Design and Analysis of a Demand Adaptive and Locality Aware Streaming Media Server Cluster" *ACM/Springer Multimedia Systems Journal*, 13(3), pages 235-249, September 2007

J52. B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer and A. Tantawi, "Analytic Modeling of Multi-tier Internet Applications," *ACM Transactions on the Web (TWEB)*, 1(1):1–25, May 2007.

J53. H. Li, K. Ramamritham, P. Shenoy, R. Grupen, J. Sweeney, "Resource Management for Real-time Tasks in Mobile Robotics," Special Issue on Dynamic Resource Management in Distributed Real-Time Systems, *Journal of Systems and Software*, 80:962-971, May 2007

J54. X. Liu, J. Lan, P. Shenoy and K. Ramamritham, "Consistency Maintenance in Dynamic Overlay Networks," *Computer Networks*, Special Issue on Overlay Distribution Structures and their Applications, 50(6), pp 859-876, April 2006.

J55. A. Chandra, P. Pradhan, R. Tewari, S. Sahu and P. Shenoy, "An Observation-based Approach Towards Self-managing Web Servers" *Computer Communications*, 2006.

J56. Y. Guo, Z. Ge, B. Urgaonkar, P. Shenoy and D. Towsley, "Dynamic Cache Reconfiguration Strategies for Cluster-based Streaming Proxies," *Computer Communications*, 29(8), pp 1174-1188, May 2006.

J57. J. Arnold, B.N. Levine, R. Manmatha, F. Lee, P. Shenoy, M.-C. Tsai, T.K. Ibrahim, D. OBrien, D.A. Walsh, "Information Sharing in Out-of-Hospital Disaster Response: The future role of information technology", *Journal of Pre-Hospital and Disaster Medicine*, 19(2):201-207, Sept 2004.

J58. Sheetal Shah, Krithi Ramamritham and Prashant Shenoy, "Resilient and Coherency Preserving Dissemination of Dynamic Data Using Cooperating Peers," *IEEE Trans. on Knowledge and Data Engineering, Special Issue on Peer-to-Peer Data Management*, 16(7):799-812 July 2004.

J59. Aameek Singh, Abhishek Trivedi, Krithi Ramamritham and Prashant Shenoy, "PTC : Proxies that Transcode and Cache in Heterogeneous Web Client Environments," *World Wide Web Journal, Special Issue on WISE 2002 papers*, 7(1):2-28 2004.

J60. Bhuvan Urgaonkar and Prashant Shenoy, "Sharc: Managing CPU and Network Bandwidth in Shared Clusters," *IEEE Trans. on Parallel and Distributed Systems (TPDS)*, 15(1):2-17, January 2004.

J61. P Shenoy, P Goyal, S Rao and H Vin, "Design Considerations for the Symphony Integrated File System", *ACM/Springer Multimedia Systems Journal*, volume 9, no 4, November 2003

J62. M. Bradshaw, B. Wang, S. Sen, L. Gao, J. Kurose, P. Shenoy, and D. Towsley, "Periodic Broadcast and Patching Services: Implementation, Measurement and Analysis in an Internet Streaming Media Testbed," *ACM/Springer Multimedia Systems Journal, Special Issue on Multimedia Distribution,* 9(1), pages 78-93, July 2003.

J63. Prashant Shenoy, Pawan Goyal, Sriram Rao and Harick Vin, "Design Considerations for the Symphony Integrated Multimedia File System," *ACM/Springer Multimedia Systems Journal*, 9(4), pages 337- 352, October 2003.

J64. Venkata Duvvuri, Prashant Shenoy and Renu Tewari, "Adaptive Leases: A Strong Consistency Mechanism for the World Wide Web," *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 15(5), pages 1266-1276, September 2003.

J65. Anoop Ninan, Purushottam Kulkarni, Prashant Shenoy, Krithi Ramamritham and Renu Tewari, "Scalable Consistency Maintenance in Content Distribution Networks using Cooperative Leases," *IEEE Transactions of Knowledge and Data Engineering*, 15(4), pages 813-828, July/August 2003.

J66. Pavan Deolasee, Amol Katkar, Ankur Panchbudhe, Krithi Ramamritham and Prashant Shenoy "Adaptive Push-Pull of Dynamic Web Data," *IEEE Transactions on Computers*, 51(6) pages 652-668, June 2002.

J67. Mohammad S. Raunak, Prashant Shenoy, Pawan Goyal, Krithi Ramamritham and Puru Kulkarni, "Implications of Proxy Caching for Provisioning Servers and Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, Special Issue on Internet Proxy Services, 20(7), pages 1276-1289, September 2002.

J68. Prashant Shenoy, Pawan Goyal and Harrick M. Vin, "Architectural Considerations for Next Generation File Systems", *ACM/Springer Multimedia Systems Journal*, 8(4), pages 270-283, July 2002

J69. Prashant Shenoy and Harrick M. Vin, "Cello: A Disk Scheduling Framework for Next Generation Operating Systems," *Real Time Systems Journal: Special Issue on Flexible Scheduling of Real-Time Systems*, 22(1) pages 9-47, January 2002.

J70. Prashant J. Shenoy and Harrick M. Vin, "Failure Recovery Algorithms for Multimedia Servers," *ACM/Springer Multimedia Systems Journal*, 8(1) pages 1-19, January 2000.

J71. Prashant J. Shenoy and Harrick M. Vin, "Efficient Striping Techniques for Variable Bit-rate Continuous Media File Servers," *Performance Evaluation Journal*, 38:(3), pages 175-200, December 1999.

J72. Prashant J. Shenoy and Harrick M. Vin, "Efficient Support for Interactive Operations in Multi-resolution Video Servers," *ACM/Springer Multimedia Systems Journal*, 7(3), pages 241–251, July 1999.

J73. Prashant J. Shenoy, Pawan Goyal, and Harrick M. Vin, "Issues in Multimedia Server Design," *ACM Computing Surveys (Special Issue : Symposium on Multimedia Systems)*, 27(4), pages 636–639, December 1995.

Refereed Conference and Workshop Publications

C1. Pradeep Ambati, Noman Bashir, David Irwin, and Prashant Shenoy, "Waiting Game: Optimally Provisioning Fixed Resources for Cloud-enabled Schedulers," Proceedings of ACM/IEEE Conference on High Performance Computing, Networking, Storage and Analysis (Supercomputing 2020 - SC20), November 2020.

C2. Qianlin Liang, Prashant Shenoy, David Irwin, "AI on the Edge: Rethinking AI-based IoT Applications Using Specialized Edge Architectures," Proceedings of IEEE International Symposium on Workload Characterization, October 2020.

C3. Amee Trivedi, Phuthipong Bovornkeeratiroj, Joseph Breda, Prashant Shenoy, Jay Taneja, and David Irwin, "Phone-based Ambient Temperature Sensing Using Opportunistic Crowdsensing and Machine Learning," Proceedings of 11th International Green and Sustainable Computing Conference (IGSC 2020), October 2020.

C4. Noman Bashir, Prashant Shenoy, David Irwin, "A Probabilistic Approach to Committing Solar Energy in Day-ahead Electricity Markets," Proceedings of 11th International Green and Sustainable Computing Conference (IGSC 2020), October 2020.

C5. Pradeep Ambati, David Irwin and Prashant Shenoy, "A First Look at Amazon's Reserved Instance Marketplace," Proceedings of 12th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '20), Boston, MA, July 2020.

C6. Alex Fuerst, Ahmed Ali-Eldin, Prashant Shenoy and Prateek Sharma, "Cloud-scale VM-deflation for Running Interactive Applications On Transient Servers," Proceedings of ACM Symposium on High-Performance Parallel and Distributed Computing, Stockholm, Sweden, June 2020.

C7. Akhil Soman, Amee Trivedi, David Irwin, Beka Kosanovic, Ben McDaniel, and Prashant Shenoy, "Peak Forecasting for Battery-based Energy Optimizations in Campus Microgrids," Proceedings of ACM Intl. Conference on Future Energy Systems (e-Energy), Melbourne, Australia, June 2020.

C8. Rishikesh Jha, Stephen Lee, Srinivasan Iyengar, Mohammad Hajiesmaili, David Irwin and Prashant Shenoy, "Emission-aware Energy Storage Scheduling for a Greener Grid," Proceedings of ACM Intl. Conference on Future Energy Systems (e-Energy), Melbourne, Australia, June 2020.

C9. Menghong Feng, Noman Bashir, David Irwin, Prashant Shenoy and Dragoljub Kosanovic, "SunDown: Model-driven Per-Panel Solar Anomaly Detection for Residential Arrays," Proceedings of ACM Conference on Computing and Sustainable Societies (COMPASS), Guayaquil, Ecuador, June 2020.

C10. Pradeep Ambati, Noman Bashir, David Irwin, Mohammad Hajiesmaili and Prashant Shenoy, "Hedge Your Bets: Optimizing Long-term Cloud Costs by Mixing VM Purchasing Options," Proceedings of IEEE International Conference on Cloud Engineering (IC2E), Sydney, April 2020 .

C11. Phuthipong Bovornkeeratiroj, Srinivasan Iyengar, Stephen Lee, David Irwin and Prashant Shenoy "RepEL: A Utility-preserving Privacy System for IoT-based Energy Meters," Proceedings of ACM/IEEE Conference on Internet of Things Design and Implementation (IoTDI), Sydney, April 2020 .

C12. Joseph Breda, Amee Trivedi, Chulabhaya Wijesundara, Phuthipong Bovornkeeratiroj, David Irwin, Prashant Shenoy and Jay Taneja, "Hot or Not: Leveraging Mobile Devices for Ubiquitous Temperature Sensing," Proceedings of ACM BuildSys, New York, NY, November 2019.

C13. Noman Bashir, Dong Chen, David Irwin, and Prashant Shenoy, "Solar-TK: A Data-driven Toolkit for Solar PV Performance Modeling and Forecasting," Proceedings of IEEE Mobile Ad-hoc and Sensor Systems (MASS 2019), Monterey, CA , November 4-7 2019.

C14. Stephen Lee, Srinivasan Iyengar, Menghong Feng, Prashant Shenoy, Subhransu Maji, "DeepRoof: A Data-driven Approach For Solar Potential Estimation for Rooftop Imagery," Proceedings of 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2019), Anchorage, Alaska, August 2019.

C15. David Irwin, Prashant Shenoy, Pradeep Ambati, Prateek Sharma, Supreeth Shastri and Ahmed Ali-Eldin, "The Price is (Not) Right: Reflections on Pricing for Transient Cloud Servers," Proceedings of IEEE ICCCN Valencia, Spain, July 2019.

C16. Vani Gupta, Prashant Shenoy and Ramesh Sitaraman, "Combining Renewable Solar and Open Air Cooling for Internet-scale Distributed Networks," Proceedings of 10th ACM Conference on Future Energy Systems (ACM e-Energy), Phoenix, AZ, June 2019.

C17. Pradeep Ambati, David Irwin, Prashant Shenoy, Lixin Gao, Ahmed Ali-Eldin, and Jeannie Albrecht, "Understanding the Synchronization Costs of Distributed ML on Transient Cloud Resources," Proceedings of IEEE Intl Cloud Engg Conference (IC2E), June 2019.

C18. Prateek Sharma, Ahmed Ali-Edlin, Prashant Shenoy, "Resource Deflation: A New Approach For Transient Resource Reclamation," Proceedings of ACM Eurosys, Dresden, Germany, March 2019.

C19. Ahmed Ali-Eldin, Jonathan Westin, Bin Wang, Prateek Sharma, Prashant Shenoy "SpotWeb: Running Latency-sensitive Distributed Web Services on Transient Cloud Servers," Proceedings of the 28th ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC), Phoenix, AZ, June 2019

C20. Lucas Chaufournier, Ahmed Ali-Eldin, Prateek Sharma, Don Towsley and Prashant Shenoy, "Performance Evaluation of Multi-Path TCP For Data Center and Cloud Workloads," Proceedings of ACM/SPEC International Conference on Performance Engineering (ICPE 2019), Mumbai, India, April 7-11, 2019

C21. Noman Bashir, David Irwin, and Prashant Shenoy, "Helios: A Programmable Software-defined Solar Module," Proceedings of the ACM International Conference on Systems for Energy Efficient Built Environments (Buildsys 2017), Shenzhen, China, November 2018.

C22. Srinivasan Iyengar, Stephen Lee, David Irwin, Prashant Shenoy and Benjamin Weil, "WattHome: Identifying Energy-Inefficient Homes at City-scale," *Proceedings of 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2018)*, London, August 2018

C23. John Wamburu, Stephen Lee, Prashant Shenoy and David Irwin "Analyzing Distribution Transformers at City Scale and the Impact of EVs and Storage," *Proceedings of 9th ACM Conference on Future Energy Systems (ACM e-Energy)*, Karlsruhe, Germany, June 2018.

C24. Stephen Lee, Prashant Shenoy, Krithi Ramamritham, David Irwin "vSolar: Virtualizing Community Solar and Storage for Energy Sharing," *Proceedings of 9th ACM Conference on Future Energy Systems (ACM e-Energy)*, Karlsruhe, Germany, June 2018.

C25. Srinivasan Iyengar, Stephen Lee, Daniel Sheldon, and Prashant Shenoy, "SolarClique: Detecting Anomalies in Residential Solar Arrays," *Proceedings of the First ACM Conference on Computing and Sustainable Societies (COMPASS 2018)*, Menlo Park, CA, June 2018.

C26. Sandhya Saisubramanian , Shlomo Zilberstein, and Prashant Shenoy, "Planning Using a Portfolio of Reduced Models with Cost Adjustments," *Proceedings of 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2018)*, Stockholm, Sweden, July 2018.

C27. T Abdelzaher, N Ayanian, T Basar, S Diggavi, J Diesner, D Ganesan, R Govindan, S Jha, T Lepoint, B Marlin, K Nahrstedt, D Nicol, R Rajkumar, S Russell, S Seshia, F Sha, P Shenoy, M Srivastava, G Sukhatme, A Swami, P Tabuada, D Towsley, N Vaidya, and V Veeravalli, "Can Distributed Computing Revolutionize Peace? An IoBT Perspective," *Proceedings of the 38th IEEE International Conference on Distributed Computing Systems,* Vienna, Austria, July 2018.

C28. Dong Chen, Phuthipong Bovornkeeratiroj, David Irwin and Prashant Shenoy, "Private Memoirs of IoT Devices: Safeguarding User Privacy in the IoT Era," *Proceedings of the 38th IEEE International Conference on Distributed Computing Systems,* Vienna, Austria, July 2018.

C29. Vani Gupta, Prashant Shenoy and Ramesh Sitaraman, "Efficient Solar Provisioning for Net-Zero Internet-Scale Distributed Networks," *Proceedings of the Tenth IEEE Intl. Conference on Communication Systems and Networks (COMSNETS)*, Bangalore, India, January 2018.

C30. Noman Bashir, David Irwin, Prashant Shenoy, and Jay Taneja, "Enforcing Fair Grid Energy Access for Controllable Distributed Solar Capacity," *Proceedings of the ACM International Conference on Systems for Energy Efficient Build Environments (Buildsys 2017)*, Delft, The Netherlands, November 2017

C31. Haroon Rashid, Priyanka Mary Mammen, Siddharth Singh, Krithi Ramamritham, Pushpendra Singh, and Prashant Shenoy "Want to reduce Energy Consumption? Don't depend on the Customers," *Proceedings of the ACM International Conference on Systems for Energy Efficient Build Environments (Buildsys 2017)*, Delft, The Netherlands, November 2017

C32. Lucas Chaufournier, Prateek Sharma, Franck Le, Erich Nahum, Prashant Shenoy and Don Towsley "Fast Transparent Virtual Machine Migration in Distributed Edge Clouds," *The Second ACM/IEEE Symposium on Edge Computing San Jose, CA*, October 2017

C33. David Irwin, Prateek Sharma, Supreeth Shastri, and Prashant Shenoy, "The Financialization of Cloud Computing: Opportunities and Challenges" *Proceedings of IEEE ICCCN* 2017.

C34. Prateek Sharma, David Irwin, and Prashant Shenoy, "Portfolio-driven Resource Management for Transient Cloud Servers," *Proceedings of ACM SIGMETRICS*, Urbana, IL, June 2017.

C35. Stephen Lee, Srinivasan Iyengar, David Irwin, and Prashant Shenoy, "Distributed Rate Control for Smart Solar-powered Systems," *Proceedings of ACM Conference on Future Energy Systems (ACM e-Energy)*, Hong Kong, May 2017.

C36. Amee Trivedi, Jeremy Gummeson, Prashant Shenoy, Deepak Ganesan, and David Irwin "iSchedule: Campus-scale HVAC Schedule via Mobile WiFi Monitoring," *Proceedings of ACM Conference on Future Energy Systems (ACM e-Energy)*, Hong Kong, May 2017.

C37. Akansha Singh, Stephen Lee, David Irwin, and Prashant Shenoy, "SunShade: Enabling Software-defined Solar-powered Systems," *Proceedings of the ACM Conference on Cyber-Physical Systems (IC-CPS)*, Pittsburgh, PA, April 2017.

C38. Prateek Sharma, Lucas Chaufournier, Prashant Shenoy, and Y.C. Tay, "Containers and Virtual Machines at Scale: A Comparative Study," *Proceeding of ACM Middleware Conference*, Trento, Italy, December 12-16, 2016.

C39. Dong Chen, Srinivasan Iyengar, David Irwin, and Prashant Shenoy "SunSpot: Exposing the Location of Anonymous Solar-powered Homes," *Proceedings of the ACM International Conference on Systems for Energy Efficient Build Environments (Buildsys 2016)*, Palo Alto, CA, November 16-17, 2016

C40. Srinivasan Iyengar, Stephen Lee, David Irwin, and Prashant Shenoy "Analyzing Energy Usage on a City-scale using Utility Smart Meters," *Proceedings of the ACM International Conference on Systems for Energy Efficient Build Environments (BuildSys 2016)*, Palo Alto, CA, November 16-17, 2016

C41. Stephen Lee, Srinivasan Iyengar, David Irwin, and Prashant Shenoy, "Shared Solar-powered EV Charging Stations: Feasibility and Benefits," *Proc of IEEE 7th International Green and Sustainable Computing Conference*, Hangzhou, China, November 7-9, 2016

C42. Dong Chen, David Irwin, and Prashant Shenoy "SmartSim: A Device-accurate Smart Home Energy Trace Generator," *Proceedings of the IEEE International Conference on Smart Grid Communications*, Sydney, Australia, November 6-9, 2016

C43. Xin He, Tian Guo, Erich Nahum, and Prashant Shenoy, "Placement Strategies for Virtualized Network Functions in a NFaaS Cloud," *Proceeding of IEEE Workshop on Hot Topics in Web Systems and Technologies,* Washington DC, October 24-25, 2016

C44. Xin He and Prashant Shenoy, "Firebird: Network-aware Task Scheduling for Spark Using SDNs," *Proceedings of the 25th IEEE International Conference on Computer Communication Networks*, Waikoloa, Hawaii, August 1-4, 2016

C45. Srinivasan Iyengar, David Irwin, and Prashant Shenoy, "Non-Intrusive Model Detection: Automated Modeling of Residential Electrical Loads," *Proceedings of the ACM International Conference on Future Energy Systems (e-Energy '16)*, Waterloo, Canada, June 21-24, 2016

C46. Vani Gupta, Stephen Lee, Prashant Shenoy, Ramesh K. Sitaraman, and Rahul Urgaonkar, "How to Cool Internet-Scale Distributed Networks on the Cheap," *Proceedings of the ACM International Conference on Future Energy Systems (e-Energy '16)*, Waterloo, Canada, June 21-24, 2016

C47. Prateek Sharma, David Irwin, and Prashant Shenoy, "How Not to Bid the Cloud," *Proceedings of the 8th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '16)*, Denver, Colorado, June 20-21, 2016

C48. Xue Ouyang, David Irwin, and Prashant Shenoy, "SpotLight: An Information Service for the Cloud," *Proceedings of the 36th IEEE International Conference on Distributed Computing Systems (ICDCS '16)*, Nara Hotel, Nara, Japan, June 27-30, 2016.

C49. Prateek Sharma, Tian Guo, Xin He, David Irwin, and Prashant Shenoy, "Flint: Batch-Interactive Data-Intensive Processing on Transient Servers," *Proceedings of the Eleventh European Conference on Computer Systems (EuroSys '16)*, London, UK, April 18-21, 2016

C50. Tian Guo, Prashant Shenoy, and Hakan Hacigumus, "GeoScale: Providing Geo-Elasticity in Distributed Clouds," *Proceedings of the 2016 IEEE International Conference on Cloud Engineering (IC2E '16)*, Berlin, Germany, April 4-8, 2016

C51. Supreeth Subramanya, Zain Mustafa, David Irwin, and Prashant Shenoy, "Beyond Energy-Efficiency: Evaluating Green Datacenter Applications for Energy-Agility," *Proceedings of the 7th ACM/SPEC International Conference on Performance Engineering (ICPE 2016)*, Delft, the Netherlands, March 12-16, 2016

C52. Patrick Pegus II, Benoy Varghese, Tian Guo, David Irwin, Prashant Shenoy, Anirban Mahanti, James Culbert, John Goodhue, and Chris Hill, "Analyzing the Efficiency of a Green University Data Center," *Proceedings of the 7th ACM/SPEC International Conference on Performance Engineering (ICPE 2016)*, Delft, the Netherlands, March 12-16, 2016

C53. Prateek Sharma, Purushottam Kulkarni, and Prashant Shenoy, "Per-VM Page Cache Partitioning for Cloud Computing Platforms," *Proceedings of IEEE International Conference on Communication Systems and Networks*, Bangalore, India, January 5-9, 2016

C54. Srinivasan Iyengar, Sandeep Kalra, Anushree Ghosh,David Irwin, Prashant Shenoy, and Benjamin Marlin, "iProgram: Inferring Smart Schedules for Dumb Thermostats ," *Proceedings of the 2nd ACM International Conference on Embedded Systems For Energy-Efficient Built Environments (BuildSys'15)*, Stanford, California, November 16-17, 2015.

C55. Pengcheng Xiong, Xin He, Hakan Hacigumus, and Prashant Shenoy, "Cormorant: Running Analytic Queries on MapReduce with Collaborative Software-defined Networking," *Proceedings of Third IEEE Workshop on Hot Topics in Web Systems and Technologies*, Washington DC, November 2015

C56. Boduo Li, Yanlei Diao, Prashant Shenoy, "Supporting Scalable Analytics with Latency Constraints," *Proceedings of the 2015 VLDB conference*, Hawaii, August 2015. Also appears in PVLDB Journal, 8(11), 2015.

C57. Supreeth Subramanya, Tian Guo, Prateek Sharma, David Irwin, and Prashant Shenoy, "SpotOn: A Batch Computing Service for the Spot Market," *Proceedings of 6th ACM Symposium on Cloud Computing (SOCC)*, Hawaii, August 2015

C58. Aditya Mishra, Ramesh Sitaraman, David Irwin, Ting Zhu, Prashant Shenoy, Bhavana Dalvi, and Stephen Lee, "Integrating Energy Storage in Electricity Distribution Networks," *Proceedings of the 6th ACM Intl. Conference on Future Energy Systems (ACM e-Energy)*, Bangalore, India, July 2015

C59. Tian Guo and Prashant Shenoy, "Model-driven Geo-Elasticity In Database Clouds," *Proceedings of the 12th IEEE International Conference on Autonomic Computing (ICAC)*, July 2015.

C60. Stephen Lee, Rahul Urgaonkar, Ramesh Sitaraman and Prashant Shenoy, "Cost Minimization using Renewable Cooling and Thermal Energy Storage in CDNs," *Proceedings of the 12th IEEE International Conference on Autonomic Computing (ICAC)*, July 2015

C61. Xin He, Ramesh Sitaraman, Prashant Shenoy, and David Irwin, "Cutting the Cost of Hosting Online Internet Services using Cloud Spot Markets," *Proceedings of the 24th international symposium on High-performance parallel and distributed computing*, Portland, Oregon, June 15-19, 2015

C62. Vani Gupta, Stephen Lee, Rahul Urgaonkar, Ramesh Sitaraman and Prashant Shenoy "Towards cooling Internet-Scale Distributed Networks on the Cheap," (poster paper) *Proceedings of the ACM Sigmetrics Conference*, Portland, Oregon, June 2015.

C63. Prateek Sharma, Stephen Lee, Tian Guo, David Irwin, and Prashant Shenoy, "SpotCheck: Designing a Derivative IaaS Cloud on the Spot Market," *Proceedings of the Tenth European Conference on Computer Systems*, EuroSys '15

C64. Patrick Pegus II, Emmanuel Cecchet, and Prashant Shenoy, "Video BenchLab: An Open Platform for Realistic Benchmarking of Streaming Media Workloads," *Proceedings of the ACM Multimedia Systems Conference (MMSys 2015)*, Portland, OR, March 2015.

C65. Sean Barker, Sandeep Kalra, David Irwin, and Prashant Shenoy, "PowerPlay: Creating Virtual Power Meters through Online Load Tracking in Smart Homes," Proceedings of the ACM Intl. Conference on Embedded Systems for Energy-efficient Buildings (BuildSys 2014), Memphis, TN, November 2014.

C66. Srini Iyengar, Navin Sharma, David Irwin, Prashant Shenoy, and Krithi Ramamritham, "SolarCast - A Cloud-based Black Box Solar Predictor for Smart Homes," *Proceedings of the ACM Intl. Conference on Embedded Systems for Energy-efficient Buildings (BuildSys 2014)*, Memphis, TN, November 2014.

C67. Z. Huang, J. Su, T. Zhu, A. Sharma, A. Ambegaonkar, Y. Gu, A. Mishra, D. Irwin, and P. Shenoy, "Minimizing Electricity Costs by Sharing Energy in Sustainable Microgrids," *Proceedings of the ACM Intl. Conference on Embedded Systems for Energy-efficient Buildings (BuildSys 2014)*, Memphis, TN, November 2014.

C68. Sean Barker, Moaj Musthag, David Irwin, and Prashant Shenoy, "Non-Intrusive Load Identification for Smart Outlets," *Proceedings of IEEE Intl. Conf on Smart Grid Communications (SmartGridComm 2014)*, Pisa, Italy, November 2014.

C69. B. Varghese, N. Carlsson, G. Jourjon, A. Mahanti, and P. Shenoy, "Greening Web Servers: A Case for Ultra Low Power Web Servers," *Proceedings of the 5th IEEE Intl. Green Computing Conference (IGCC)*, Dallas, TX, November 2014.

C70. Sean Barker, Yun Chi, Hakan Hacigumus, Prashant Shenoy and Emmanuel Cecchet, "ShuttleDB: Database-Aware Elasticity in the Cloud," *Proceedings of USENIX Intl. Conference on Autonomic Computing (ICAC)*, Philadeplhia, PA, June 2014.

C71. Sean Barker, Sandeep Kalra, David Irwin, and Prashant Shenoy, "NILM Redux: The Case for Emphasizing Applications over Accuracy," *Proceedings of the Second NILM workshop (NILM 2014)*, Austin, TX, June 2014.

C72. Dong Chen, David Irwin, Prashant Shenoy, and Jeannie Albrecht, "Combined Heat and Privacy: Preventing Occupancy Detection from Smart Meters," *Proceedings of IEEE PerCom*, Budapest, Hungary, March 2014.

C73. Tian Guo, Vijay Gopalakrishnan, K. K. Ramakrishnan, Prashant Shenoy, Arun Venkataramani, Seungjoon Lee, "VMShadow: Optimizing The Performance of Latency-sensitive Virtual Desktops in Distributed Clouds," *Proceedings of ACM Multimedia Systems 2014 (MMSYS 2014)*, Singapore, March 19-21 2014

C74. Vimal Mathew, Ramesh K. Sitaraman, Prashant Shenoy, "Reducing Energy Costs in Internet-Scale Distributed Systems Using Load Shifting," *Proceedings of IEEE COMSNETS 2014*, Bangalore, India, January 2014

C75. Ye Xu, David Irwin, and Prashant Shenoy "Incentivizing Advanced Load Scheduling in Smart Homes," *Proceedings of ACM BuildSys 2013*, Rome, Italy, November 2013

C76. Dong Chen, Sean Barker, Adarsh Subbaswamy, David Irwin, and Prashant Shenoy "Non-Intrusive Occupancy Monitoring using Smart Meters," *Proceedings of ACM BuildSys*, Rome, Italy, November 2013

C77. Upendra Sharma, Prashant Shenoy and Sambit Sahu, "A Flexible Elastic Control Plane for Private Clouds," *Proceedings of the ACM Cloud and Autonomics Conference (CAC 2013)*, August 2013.

C78. V. Mathew, R. Sitaraman, and P. Shenoy, "Energy-Efficient Content Delivery Networks using Cluster Shutdown," *Proceedings of the IEEE International Green Computing Conference (IGCC)*, Arlington, VA, June 2013

C79. S. Barker, S. Kalra, D. Irwin, P. Shenoy, "Empirical Characterization and Modeling of Electrical Loads in Smart Homes," *Proceedings of the IEEE International Green Computing Conference (IGCC)*, Arlington, VA, June 2013

C80. N. Sharma, D. Irwin and P. Shenoy, "A Distributed File System for Intermittent Power," *Proceedings of the IEEE International Green Computing Conference (IGCC)*, Arlington, VA, June 2013

C81. E. Cecchet, R. Sims, X. He and P. Shenoy, "mBenchLab: Measuring the QoE of Web Applications using Mobile Devices," *Proceedings of the ACM/IEEE Intl. Symposium on Quality of Service (IWQoS)*, Montreal, Canada, June 2013 Best Paper Award

C82. A. Mishra, D. Irwin, P. Shenoy, T. Zhu, "Scaling Distributed Energy Storage for Grid Peak Reduction," *Proceedings of the 4th ACM Intl. Conference on Future Energy Systems (ACM e-Energy)*, Berkeley, CA, May 2013 (Best Paper Runner-up)

C83. Rahul Singh, David Irwin, Prashant Shenoy, and K.K. Ramakrishnan, "Yank: Enabling Green Data Centers to Pull the Plug," *Proceedings of NSDI 2013*, Lombard, IL, April 2013

C84. T. Zhu, Z. Huang, A. Sharma, J. Su, D. Irwin, A. Mishra, D. Menasche, and P. Shenoy, "Sharing Renewable Energy in Smart Microgrids," *ACM/IEEE 4th International Conference on Cyber-Physical Systems (ACM/IEEE ICCPS '13)*, Philadelphia, PA, April 2013.

C85. Navin Sharma, Dilip Krishnappa, David Irwin, Michael Zink, Prashant Shenoy, "GreenCache: Augmenting Off-the-Grid Cellular Towers with Multimedia Caches," *Proceedings of ACM MMSys*, Oslo, Norway, February 2013

C86. Upendra Sharma, Prashant Shenoy and Don Towsley, "Provisioning Multi-tier Cloud Applications Using Statistical Bounds on Sojourn Time," *Proceedings of the 9th ACM Conference on Autonomic Computing*, San Jose, CA, September 2012.

C87. Sean Barker, Aditya Mishra, David Irwin, Emmanuel Cecchet, Prashant Shenoy, and Jeannie Albrecht, "Smart: An Open Data Set and Tools for Enabling Research in Sustainable Homes," *Proceedings of the 2012 Workshop on Data Mining Applications in Sustainability (SustKDD 2012)*, Beijing, China, August 2012

C88. Aditya Mishra, David Irwin, Prashant Shenoy, Jim Kurose, and Ting Zhu, "SmartCharge: Cutting the Electricity Bill in Smart Homes with Energy Storage," *Proceedings of the Third International ACM Conference on Future Energy Systems (e-Energy)*, Madrid, Spain, May 2012.

C89. T. Guo, U. Sharma, S. Sahu, T. Wood, P. Shenoy, "Seagull: Intelligent Cloud Bursting for Enterprise Applications," *Proceedings of USENIX Annual Technical Conference*, Boston, MA, June 2012.

C90. S. Barker, T. Wood, P. Shenoy, and R. Sitaraman, "An Empirical Study of Memory Sharing in Virtual Machines," *Proceedings of the 2012 USENIX Annual Technical Conference (USENIX ATC)*, Boston, MA, June 2012.

C91. R. Singh, P. Shenoy, K. K. Ramakrishnan, R. Kelkar and H. Vin, "eTransform: Transforming Enterprise Data Centers by Automated Consolidation," *Proceedings of the 32nd International Conference on Distributed Computing Systems (ICDCS)*, Macau, China, June 2012.

C92. A. Mishra, D. Irwin, P. Shenoy, J. Kurose, and T. Zhu, "SmartCharge: Cutting the Electricity Bill in Smart Homes with Energy Storage," *Proceedings of the Third International Conference on Future Energy Systems (e-Energy)*, Madrid, Spain, May 2012.

C93. S. Barker, Y. Chi, H J. Moon, H. Hacigumus, P. Shenoy, "Cut Me Some Slack: Latency-aware Live Migration for Databases," *Proceedings of Emerging Database Technologies conference (EDBT)*, April 2012.

C94. Sean Barker, Aditya Mishra, David Irwin, Prashant Shenoy, and Jeannie Albrecht, "SmartCap: Flattening Peak Electricity Demand in Smart Homes *Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Lugano, Switzerland, March, 2012 *(Best papers of PerCom'12)*.

C95. Andres Molina-Markham, George Danezis, Kevin Fu, Prashant Shenoy and David Irwin, "Designing Privacy-preserving Smart Meters with Low-cost Microcontrollers," *Proceedings of Financial Cryptography and Data Security*, Bonaire, March 2012

C96. Vaishali Sadaphal, Maitreya Natu, Harrick Vin, and Prashant Shenoy, "Varanus: More-With-Less Fault Localization in Data Centers," *Proceedings of IEEE COMSNETS conference*, Bangalore India, January 2012 *(Best paper award).*

C97. Vimal Mathew, Ramesh Sitaraman and Prashant Shenoy, "Energy-Aware Load Balancing in Content Delivery Networks," *Proceedings of IEEE Infocom*, Orlando, FL, March 2012

C98. Rahul Singh, Prashant Shenoy, Maitreya Natu, Vaishali Sadaphal and Harrick Vin, "Predico: A System for What-if Analysis in Complex Data Center Applications," *Proceedings of the 12th ACM/IFIP/USENIX International Middleware Conferenc*e, Lisboa, Portugal, December 2011.

C99. Timothy Wood, Andres Lagar-Cavilla, K. K. Ramakrishnan, Prashant Shenoy, Jacobus Van der Merwe, "PipeCloud: Using Causality to Overcome Speed-of-Light Delays in Cloud-Based Disaster Recovery," *Proceedings of 2nd ACM Symposium on Cloud Computing (SOCC)*, October 2011.

C100. David Irwin, Anthony Wu, Sean Barker, Aditya Mishra, Jeannie Albrecht, and Prashant Shenoy, "Exploiting Home Automation Protocols for Load Monitoring in Smart Buildings," *Proceedings of the Third ACM Workshop on Embedded Sensing Systems for Energy-efficiency in Buildings (BuildSys), Seattle, WA*, November 2011.

C101. Ting Zhu, Aditya Mishra, David Irwin, Navin Sharma, Prashant Shenoy, and Don Towsley, "The Case For Efficient Renewable Energy Management for Smart Homes," *Proceedings of the Third ACM Workshop on Embedded Sensing Systems for Energy-efficiency in Buildings (BuildSys), Seattle,WA*, November 2011.

C102. Navin Sharma, Pranshu Sharma, David Irwin, and Prashant Shenoy, "Predicting Solar Generation from Weather Forecasts Using Machine Learning," *Proceedings of the Second IEEE International Conference on Smart Grid Communications (SmartGridComm)*, Brussels, Belgium, October, 2011.

C103. David Irwin, Navin Sharma, and Prashant Shenoy, "Towards Continuous Policy-driven Demand Response in Data Centers," *Proceedings of the ACM SIGCOMM Workshop on Energy and IT: from Green Networking to Smarter Systems (GreenNet)*, Toronto, Canada, August 2011 *(Best Paper Award)*.

C104. Emmanuel Cecchet, Veena Udayabhanu, Timothy Wood and Prashant Shenoy, "BenchLab: An Open Testbed for Realistic Benchmarking of Web Applications," *Proceedings of 2nd USENIX Conference on Web Application Development (WebApps '11)*, June 15-16, 2011, Portland, OR.

C105. Zhao Cao, Charles Sutton, Yanlei Diao, and Prashant Shenoy, "Distributed Inference and Query Processing for RFID Tracking and Monitoring.," *Proceedings of VLDB 2011*, Seattle, WA, August 2011

C106. Boduo Li, Ed Mazur, Yanlei Diao, Andrew McGregor, and Prashant Shenoy, "A Platform for Scalable On-pass Analytics using MapReduce," *Proceedings of ACM Sigmod, Athens*, Greece, June 2011

C107. Upendra Sharma, Prashant Shenoy, Sambit Sahu, Anees Shaikh, Kingfisher: Cost-aware Elasticity in the Cloud *Proceedings of IEEE Infocom 2011, Shanghai*, China, April 2011 (short version). Full version in the *Proceedings of the 31st IEEE ICDCS*, Minneapolis, MN, June 2011

C108. Timothy Wood, Rahul Singh, Arun Venkataramani, Prashant Shenoy and Emmanuel Cecchet, "ZZ and the Art of Practical BFT Execution," *Proceedings of EuroSys 2011*, Salzburg, Austria, April 10 - 13, 2011

C109. Boduo Li, Ed Mazur, Yanlei Diao, and Prashant Shenoy., "Towards Scalable One-Pass Analytics Using MapReduce," *Proceedings of DataCloud 2011- Workshop on Data-intensive Computing in the Clouds*, Anchorage, Alaska, May 2011

C110. Emmanuel Cecchet, Rahul Singh, Upendra Sharma, Prashant Shenoy, "Dolly: Virtualization-driven Database Provisioning for the Cloud" *Proc of the ACM International Conference on Virtual Execution Environments (VEE 2011)*, Newport Beach, CA, March 9-11, 2011.

C111. Timothy Wood, K.K. Ramakrishnan, Prashant Shenoy, Jacobus Van der Merwe, "CloudNet: Dynamic Pooling of Cloud Resources by Live WAN Migration of Virtual Machines," *Proc of the ACM International Conference on Virtual Execution Environments (VEE 2011)*, Newport Beach, CA, March 9-11, 2011.

C112. Navin Sharma, Sean Barker, David Irwin, and Prashant Shenoy, "Blink: Managing Server Clusters on Intermittent Power," *Proceedings of the Sixteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Newport Beach, California, March, 2011.

C113. Navin Sharma, David Irwin, Prashant Shenoy and Michael Zink, "MultiSense: Fine-grained Multiplexing for Steerable Camera Sensor Networks," *Proceedings of ACM Multimedia Systems Conference (MMSys 2011)*, San Jose, CA, U

C114. Andres Molina-Markham, Prashant Shenoy, Kevin Fu, Emmanuel Cecchet, David Irwin, "Private Memoirs of a Smart Meter," *Proc. of 2nd ACM Workshop On Embedded Sensing Systems For Energy-Efficiency In Buildings (BuildSys 2010)*, Zurich, Switzerland, November 2, 2010

C115. Anand Seetharam, Manikandan Somasundaram, Jim Kurose, Don Towsley and Prashant Shenoy, "Shipping to Streaming: Is this shift green?" *Proc. of First ACM SIGCOMM Workshop on Green Networking*, New Delhi, India August 2010

C116. D. Agrawal, B. Li, Z. Cao, D. Ganesan, Y. Diao and P. Shenoy, "Exploiting the Interplay Between Memory and Flash Storage In Embedded Sensor Devices," *Proceedings of the 16th IEEE Conference on Embedded and Real-time Computing Systems (RTCSA)*, Macau, China, Aug 2010

C117. Timothy Wood, Emmanuel Cecchet, K.K. Ramakrishnan, Prashant Shenoy, Jacobus van der Merwe, and Arun Venkataramani, " Disaster Recovery as a Cloud Service: Economic Benefits & Deployment Challenges," *Proceedings of the 2nd Workshop on Hot Topics in Cloud Computing (HotCloud 2010)*, June 22, 2010

C118. Rahul Singh, Upendra Sharma, Emmanuel Cecchet and Prashant Shenoy, "Autonomic Mix-Aware Provisioning for Non-Stationary Data Center Workloads," *Proceedings of the 7th IEEE International Conference on Autonomic Computing and Communications (ICAC)*, Washington, DC, USA, June 7-11, 2010.

C119. Navin Sharma, Jeremy Gummeson, David Irwin, and Prashant Shenoy, "Cloudy Computing: Leveraging Weather Forecasts in Energy Harvesting Sensor Systems," *Proceedings of the Seventh IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Boston, MA, June 2010

C120. Sean Barker and Prashant Shenoy, "Empirical Evaluation of Latency-sensitive Application Performance in the Cloud," *Proceedings of MMSys 2010 - the first ACM Multimedia Systems Conference*, Scottsdale, AZ, February 2010

C121. T. Wood, G. Tarasuk-Levin, P. Shenoy, P. Desnoyers, E. Cecchet, and M. Corner, "Memory Buddies: Exploiting Page Sharing for Smart Colocation in Virtualized Data Centers," *Proceedings of the ACM Conference on Virtual Execution Environments (VEE 2009)*, Washington DC, March 2009.

C122. Thanh Tran, Charles Sutton, Richard Cocci, Yanming Nie, Yanlei Diao and Prashant Shenoy, "Probabilistic Inference over RFID Streams in Mobile Environments," *Proceedings of the 25th International Conference on Data Engineering (ICDE)*, April 2009, Shanghai, China (to appear)

C123. Timothy Wood, Ludmila Cherkasova, Kivan Ozonat, and Prashant Shenoy, Profiling and Modeling Resource Usage of Virtualized Applications *Proceedings of the 9th ACM/Usenix conference on Middleware*, Lueven Belgium, December 2008. (18% acceptance rate).

C124. A. Nemmaluri, M. Corner, and P. Shenoy, "Sherlock: Automatically Locating Objects for Humans," *Proceedings of ACM Mobisys, Breckenridge, CO*, June 2008. (18% acceptance rate)

C125. R. Cocci, T. Tran, Y. Diao, and P. Shenoy, "Efficient Data Interpretation and Compression over RFID Streams," *Proceedings of the IEEE 24th Intl. Conference on Data Engineering (ICDE)*, April 2008, Cancun, Mexico (short paper). (31% acceptance rate)

C126. David Yates, Erich Nahum, James F. Kurose, and Prashant Shenoy, "Data Quality and Query Cost in Wireless Sensor Networks," *Proceedings of the IEEE Intl. Conference on Pervasive Computing and Communications (PerCom 2008)*, Kowloon, Hong Kong, March 2008. Invited to a fast-track special issue of Pervasive and Mobile Computing journal (11.9% acceptance rate).

C127. D. Yates, E. Nahum, J. Kurose and P. Shenoy, "Data Quality and Query Cost in Wireless Sensor Networks (short version)," *Proceedings of the Third IEEE Intl. Workshop on Sensor Networks and Systems for Pervasive Computing (PerSens 2007)*, White Plains, NY 2007 (28.9% acceptance rate).

C128. M. Li, T. Yan, D. Ganesan, E. Lyons, P. Shenoy, A. Venkataramani and M. Zink, "Multi-user Data Sharing in Radar Sensor Networks," *Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems (Sensys 2007),* Sydney, Australia, Nov 2007 (16.8% acceptance rate).

C129. K K. Ramakrishnan, P. Shenoy and Jacobus van der Merwe, "Live Data Center Migration across WANs: A Robust Cooperative Context-aware Approach," *Proceedings of the ACM SIGCOMM Workshop on Internet Network Management (INM)*, Kyoto, Japan, August 2007 (30% acceptance rate).

C130. Peter J. Desnoyers and Prashant Shenoy, "Hyperion: High Volume Stream Archival for Retrospective Querying," *Proceedings of the 2007 USENIX Annual Technical Conference, Santa Clara CA*, June 17-22 2007 (**Best Paper Award**, 19% acceptance rate).

C131. T. Wood, P. Shenoy, A. Venkataramani and M. Yousif, "Black-box and Gray-box Strategies for Virtual Machine Migration," *Proceedings of the Fourth Symposium on Networked Systems Design and Implementation (NSDI)*, Cambridge, MA, April 2007 (23.8% acceptance rate).

C132. P. Kulkarni, D. Ganesan and P. Shenoy, "Approximate Initialization of Camera Sensor Networks," *Proceedings of the 4th European Conference on Wireless Sensor Networks (EWSN 2007)*, Delft, Netherlands, January, 2007 (13.4% acceptance rate).

C133. Y. Diao, D. Ganesan, G. Mathur and P. Shenoy, "Re-thinking Data Management for Storage-centric Sensor Networks," *Proceedings of the Third Biennial Conference on Innovative Data Systems Research (CIDR)*, Asilomar, CA, January 2007 (45% acceptance rate).

C134. X. Liu, G. Niv, K. K. Ramakrishnan, P. Shenoy and J. Van der Merwe, "The Case for Semantic Aware Remote Replication," *Proc. 2nd International Workshop on Storage Security and Survivability (StorageSS 2006)*, Alexandria, VA, October 2006.

C135. X. Liu, P. Kulkarni, P. Shenoy and D. Ganesan, "Snapshot: A Self-Calibration Protocol for Camera Sensor Networks," *Proc. IEEE Workshop on Broadband Advanced Sensor Networks (BASENETS 2006)*, San Jose, CA, October 2006.

C136. G. Mathur, P. Desnoyers, D. Ganesan and P. Shenoy, "CAPSULE: An Energy-Optimized Object Storage System for Memory-Constrained Sensor Devices," *Proceedings of the Fourth ACM Conference on Embedded Networked Sensor Systems (SenSys), Boulder, CO*, November 2006 (19% acceptance rate).

C137. Xiaotao Liu, Mark Corner and Prashant Shenoy, "Ferret: RFID Localization for Pervasive Multimedia," *Proceedings of the Eight International Conference on Ubiquitous Computing (Ubicomp)*, Orange County, CA, September 2006 (13% acceptance rate).

C138. V. Sundaram, T. Wood and P. Shenoy, "Efficient Data Migration for Load Balancing in Self-managing Storage Systems," *Proceedings of the Third IEEE Intl. Conference on Autonomic Computing, Dublin*, June 2006 (21% acceptance rate).

C139. M. Li, D. Ganesan and P. Shenoy, "PRESTO: Feedback-driven Data Management in Sensor Networks," *Proceedings of the Third Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA, May 2006 (25.4% acceptance rate).

C140. G. Mathur, P. Desnoyers, D. Ganesan, P. Shenoy, "Ultra-Low Power Storage for Sensor Networks," *Proceedings of the IEEE/ACM Conference on Information Processing in Sensor Networks - SPOTS Track*, Nashville, TN, April 2006 (25.8% acceptance rate).

C141. X. Liu, M. Corner and P. Shenoy, "SEVA: Sensor Enhanced Video Annotation," *Proceedings of ACM Multimedia*, Singapore, November 2005 (**Best Paper Award**, 16% acceptance rate).

C142. P. Kulkarni, P. Shenoy and D. Ganesan, "SensEye: A Multi-tier Camera Sensor Network," *Proceedings of ACM Multimedia*, Singapore, November 2005 (**Best Student Paper Finalist**, 16% acceptance rate).

C143. X. Liu, P. Shenoy and M. Corner, "Chameloen: Application Controlled Power Management with Performance Isolation," *Proceedings of ACM Multimedia*, Singapore, November 2005 (16% acceptance rate).

C144. P. Desnoyers, D. Ganesan and P. Shenoy, "TSAR: A Two Tier Sensor Storage Architecture Using Interval Skip Graphs," *Proceedings of ACM SenSys*, San Diego, CA, November 2005 (17% acceptance rate).

C145. P. Desnoyers, D. Ganesan, H. Li and P. Shenoy, "PRESTO: A Predictive Storage Architecture for Sensor Networks," *Proceedings of Tenth Workshop on Hot Topics in Operating Systems (HotOS-X)*, Santa Fe, NM, June 2005 (19.8% acceptance rate).

C146. P. Kulkarni, D. Ganesan, and P. Shenoy, "The Case for Multi-tier Camera Sensor Networks," *Proceedings of ACM Nossdav*, Skamania, WA, June 2005 (37% acceptance rate).

C147. M. Bradshaw, J. Kurose, P. Shenoy and D. Towsley, "Online Scheduling in Modular Multimedia Systems with Stream Reuse," *Proceedings of ACM Nossdav*, Skamania, WA, June 2005 (37% acceptance rate).

C148. B. Urgaonkar, P. Shenoy, A. Chandra and P. Goyal, "Dynamic Provisioning for Multi-tier Internet Applications," *Proceedings of the 2nd IEEE Intl. Conference on Autonomic Computing (ICAC-05)*, Seattle, WA, June 2005 (**Best student paper award**, 16% acceptance rate).

C149. B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer and A Tantawi, "An Analytical Model for Multi-tier Internet Services and Its Applications," *Proceedings of the ACM Sigmetrics Conference*, Banff, Canada, June 2005 (13% acceptance rate).

C150. B. Urgaonkar and P. Shenoy, "Cataclysm: Handling Extreme Overloads in Internet Applications," *Proceedings of the Fourteenth International World Wide Web Conference (WWW 2005)*, Chiba, Japan, May 2005 (12% acceptance rate).

C151. H Li, P Shenoy and K Ramamritham, "Scheduling Messages with Deadlines in Multi-hop Real-time Sensor Networks," *Proceedings of the Eleventh IEEE Real-time and Embedded Technology and Applications Symposium (RTAS)*, San Francisco, CA, March 2005 (33% acceptance rate).

C152. M. Bradshaw, J. Kurose, L. Page, P. Shenoy and D. Towsley, "A Reconfigurable on-the-Fly Resource-aware Streaming Pipeline Scheduler," *Proceedings of the 12th Multimedia Computing and Networking Conference (MMCN)*, San Jose, CA, January 2005 (22% acceptance rate).

C153. Bhuvan Urgaonkar, Arnold Rosenberg and Prashant Shenoy, "Application Placement on a Cluster of Servers," *Proceedings of the 17th Intl. Conference of Parallel and Distributed Computing Systems*, San Francisco, CA, September 2004 (33.5% acceptance rate).

C154. Bing Wang, Jim Kurose, Prashant Shenoy and Don Towsley, "Multimedia Streaming via TCP: An Analytic Performance Study," *Proceedings of ACM Sigmetrics*, New York, NY, June 2004 (short paper). Full version appears in the *Proceedings of ACM Multimedia, New York, NY*, October 2004 (16.6% acceptance rate).

C155. John Sweeney, Rod Grupen and Prashant Shenoy, "Active QoS Flow Maintenance in Controlled Mobile Networks," *Proceedings of the Fourth IEEE International Symposium on Robotics and Automation (ISRA)*, Queretaro, Mexico, August 2004.

C156. Bhuvan Urgaonkar and Prashant Shenoy, "Cataclysm: Handling Extreme Overloads in Internet Services (brief announcement)" *Proceedings of ACM Principles of Distributed Computing (PODC)*, St Johns, New Foundland, Canada, July 2004 (23% acceptance rate).

C157. Xiaotao Liu, Prashant Shenoy and Weibo Gong, "A Time Series-based Approach for Power Management in Mobile Processors and Disks," *Proceedings of the 14th ACM Workshop on Network and Operating System Support for Audio and Video (NOSSDAV)*, Kinsale, Ireland, pages 74-81, June 2004 (25.2% acceptance rate).

C158. Xiaolan Zhang, Michael K. Bradshaw, Yang Guo, Bing Wang, Jim Kurose, Prashant Shenoy, Don Towsley, "AMPS: A Flexible, Scalable Proxy Testbed for Implementing Streaming Services," *Proceedings of the 14th ACM Workshop on Network and Operating System Support for Audio and Video (NOSSDAV)*, Kinsale, Ireland, pages 116-121, June 2004 (25.2% acceptance rate).

C159. Huan Li, Prashant Shenoy and Krithi Ramamritham, "Scheduling Communication in Real-Time Sensor Applications," *Proceedings of the Tenth IEEE Real-Time/Embedded Technology and Applications Symposium (RTAS04)*, Toronto, Canada, May 2004 (30.2% acceptance rate).

C160. Peter Radkov, Li Yin, Pawan Goyal, Prasenjit Sarkar and Prashant Shenoy "A Performance Comparison of NFS and iSCSI for IP-Networked Storage," *Proceedings of the Usenix Conference on File and Storage Technologies (FAST 04)*, San Francisco, CA, March 2004 (20.6% acceptance rate).

C161. Yang Guo, Zihui Ge, Bhuvan Urgaonkar, Prashant Shenoy, Don Towsley, "Dynamic Cache Reconfiguration Strategies for Cluster-Based Streaming Proxies," *Proceedings of the Eighth International Workshop on Web Content Caching and Distribution (WCW 2003), Hawthorne, NY*, September 2003 (32% acceptance rate).

C162. Abhishek Chandra, Pawan Goyal and Prashant Shenoy, "Quantifying the Benefits of Resource Multiplexing in On-Demand Data Centers," *Proceedings of the First ACM Workshop on Algorithms and Architectures for Self-Managing Systems (Self-Manage 2003), San Diego, CA*, June 2003 *(acceptance rate: 27%)*.

C163. Jiang Lan, Xiatao Liu, Prashant Shenoy and Krithi Ramamritham, "Consistency Maintenance in Peer-to-Peer File Sharing Networks," *Proceedings of IEEE Workshop on Internet Applications (WIAPP), San Jose, CA*, pages 90-94, June 2003 *(acceptance rate: 20%)*.

C164. Vijay Sundaram and Prashant Shenoy, "A Practical Learning-based Approach for Dynamic Storage Bandwidth Allocation," *Proceedings of ACM/IEEE Intl Workshop on Quality of Service (IWQoS), Monterey, CA*, Springer LNCS volume 2707, pages 479-497, June 2003 *(acceptance rate: 30%)*.

C165. Abhishek Chandra, Weibo Gong and Prashant Shenoy, "Dynamic Resource Allocation for Shared Data Centers Using Online Measurements," *Proceedings of ACM Sigmetrics 2003*, San Diego, CA, pages 300-301, June 2003 *(acceptance rate: 18%)*. A longer version version appeared in the *Proceedings of ACM/IEEE Intl Workshop on Quality of Service (IWQoS), Monterey, CA*, Springer LNCS, volume 2707, pages 381-400, June 2003 *(acceptance rate: 30%)*.

C166. Huan Li, John Sweeney, Krithi Ramamritham, Roderic Grupen and Prashant Shenoy, "Real-Time Support for Mobile Robotics," *Proceedings of the Ninth IEEE Real-Time/Embedded Technology and Applications Symposium (RTAS03), Toronto, Canada*, pages 10-18, May 2003 *(acceptance rate: 38%)*.

C167. Purushottam Kulkarni, Weibo Gong and Prashant Shenoy, "Scalable Techniques for Memory-efficient CDN Simulations," *Proceedings of the Twelfth World Wide Web Conference (WWW-03), Budapest, Hungary*, pages 609-618, May 2003 *(acceptance rate: 12.8%)*.

C168. Purushottam Kulkarni, Prashant Shenoy and Krithi Ramamritham, "Handling Client Mobility and Intermittent Connectivity in Mobile Web Accesses," *Proceedings of the ACM Conference on Mobile Data Management (MDM 2003), Melbourne, Australia*, Springer LNCS vol 2574, pages 401-407, January 2003 *(acceptance rate: 41%)*.

C169. Prashant Shenoy and Peter Radkov, "Proxy-assisted Power-friendly Streaming to Mobile Devices," *Proceedings of the 2003 Multimedia Computing and Networking (MMCN) Conference, Santa Clara, CA*, pages 177-191, January 2003 *(acceptance rate: 42%)*.

C170. Bhuvan Urgaonkar, Prashant Shenoy and Timothy Roscoe, "Resource Overbooking and Application Profiling in Shared Hosting Platforms," *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation (OSDI), Boston, MA*, pages 239-254, December 2002 *(acceptance rate:18%)*.

C171. Shetal Shah, Krithi Ramamritham and Prashant Shenoy, "Maintaining Coherency of Dynamic Data in Cooperating Repositories," *Proceedings of the 28th Conference on Very Large Data Bases (VLDB), Hong Kong*, pages 526-537, August 2002 *(acceptance rate: 15%)*

C172. Aameek Singh, Abhishek Trivedi, Krithi Ramamritham and Prashant Shenoy, "PTC : Proxies that Transcode and Cache in Heterogeneous Web Client Environments," *Proceedings of the Third IEEE International Conference on Web Information Systems Engineering, Singapore*, pages 11-20, December 2002 (**Award Paper**, 27% acceptance rate).

C173. Prashant Shenoy, Saif Hasan, Purushottam Kulkarni and Krithi Ramamritham, "Middleware versus Native OS Support: Architectural Considerations for Supporting Multimedia Applications," *Proceedings of the IEEE Real-time Technology and Applications Symposium (RTAS'02), San Jose, CA*, pages 23-34, September 2002 *(acceptance rate: 30%)*.

C174. Anoop Ninan, Purushottam Kulkarni, Prashant Shenoy, Krithi Ramamritham and Renu Tewari, "Cooperative Leases: Scalable Consistency Maintenance in Content Distribution Networks," *Proceedings of the Eleventh World Wide Web Conference (WWW 2002), Honolulu, Hawaii*, pages 1-12, May 2002 (**Award Paper**, 16% acceptance rate).

C175. Prashant Pradhan, Renu Tewari, Sambit Sahu, Abhishek Chandra, and Prashant Shenoy, "An Observation-based Approach Towards Self-managing Web Servers," *Proceedings of ACM/IEEE Intl Workshop on Quality of Service (IWQoS), Miami Beach, FL*, pages 13-22, May 2002 *(acceptance rate: 18%)*.

C176. Zihui Ge, Ping Ji and Prashant Shenoy, "A Demand Adaptive and Locality Aware (DALA) Streaming Media Cluster Architecture," *Proceedings of ACM Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV), Miami Beach, FL*, pages 139-146, May 2002 *(acceptance rate: 31%)*.

C177. Manish Bhide, Krithi Ramamritham and Prashant Shenoy, "Maintaining Stock Portfolios Up-to-date on the Web," *Proceedings of the 12th IEEE Workshop on Research Issues in Data Engineering (RIDE '02), San Jose, CA*, pages 60-68, February 2002 *(acceptance rate: 24%)*.

C178. Anuj Maheshwari, Aashish Sharma, Krithi Ramamritham and Prashant Shenoy, "TranSquid: Transcoding and Caching Proxy for Heterogeneous E-Commerce Environments," *Proceedings of the 12th IEEE Workshop on Research Issues in Data Engineering (RIDE '02), San Jose, CA*, pages 50-59, February 2002 *(acceptance rate: 24%)*.

C179. Vijay Sundaram and Prashant Shenoy, "Bandwidth Allocation in a Self-Managing Multimedia File Server," *Proceedings of the Ninth ACM Conference on Multimedia, Ottawa, Canada*, pages 291-301, October 2001 *(acceptance rate: 15%)*.

C180. M. Bradshaw, B. Wang, S. Sen, L. Gao, J. Kurose, P. Shenoy, and D. Towsley, "Periodic Broadcast and Patching Services: Implementation, Measurement and Analysis in an Internet Streaming Media Testbed," *Proceedings of the Ninth ACM Conference on Multimedia, Ottawa, Canada*, pages 280-290, October 2001 *(acceptance rate: 15%)*.

C181. Timothy Roscoe and Prashant Shenoy, "New Resource Control Issues in Shared Clusters," *Proceedings of the Eight International Workshop on Interactive Distributed Multimedia Systems (IDMS'01), Lancaster, UK*, Springer LNCS vol 2158, pages 2-9, September 2001.

C182. Deepak Karuppiah, Zhigang Zhu, Prashant Shenoy and Ed Riseman, "A Fault-tolerant Distributed Vision System for Object Tracking in a Smart Room," *Proceedings of the IEEE Workshop on Computer Vision Systems (ICVS'01), Vancouver, Canada*, Springer LNCS vol 2095, pages 201-219, July 2001.

C183. Abhishek Chandra, Micah Adler and Prashant Shenoy, "Deadline Fair Scheduling: Bridging the Theory and Practice of Proportionate-Fair Scheduling in Multiprocessor Servers," *Proceedings of IEEE Real-time Technology and Applications Symposium (RTAS'01), Taipei, Taiwan*, pages 3-14, June 2001 *(acceptance rate: 31%)*.

C184. Pavan Deolasee, Amol Katkar, Ankur Panchbudhe, Krithi Ramamritham and Prashant Shenoy, "Adaptive Push-Pull of Dynamic Web Data: Better Resiliency, Scalability and Coherency," *Proceedings of the Tenth World Wide Web Conference (WWW-10), Hong Kong*, pages 265-274, May 2001 *(acceptance rate: 19%)*.

C185. Bhuvan Urgaonkar, Anoop Ninan, Mohammad Raunak, Prashant Shenoy and Krithi Ramamritham, "Maintaining Mutual Consistency for Cached Web Objects," *Proceedings of the 21st International Conference on Distributed Computing Systems (ICDCS-21), Phoenix, AZ*, pages 371-380, April 2001 *(acceptance rate: 31%)*.

C186. Krithi Ramamritham, Pavan Deolasee, Amol Katkar, Ankur Panchbudhe and Prashant Shenoy, "Dissemination of Dynamic Data on the Internet," *Proceedings of DNIS 2000: International Workshop on Databases in Networked Information Systems, University of Aizu, Japan*, December 2000 *(invited paper)*.

C187. Vijay Sundaram, Abhishek Chandra, Pawan Goyal, Prashant Shenoy, Jasleen Sahni and Harrick Vin, "Application Performance in the QLinux Multimedia Operating System," *Proceedings of the Eighth ACM Conference on Multimedia, Los Angeles*, CA, pages 127-136, November 2000 *(acceptance rate: 17%)*.

C188. Abhishek Chandra, Micah Adler, Pawan Goyal and Prashant Shenoy "Surplus Fair Scheduling: A Proportional-Share CPU Scheduling Algorithm for Symmetric Multiprocessors," *Proceedings of the Fourth Symposium on Operating System Design and Implementation (OSDI 2000), San Diego, CA*, pages 45-59, October 2000 *(acceptance rate: 21%)*.

C189. Prashant Shenoy, "The Case for Reexamining Integrated File System Design," *Proceedings of the Tenth International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'00), Chapel Hill, NC*, pages 51-54, June 2000 *(acceptance rate: 41%)*.

C190. Mohammad S. Raunak, Prashant Shenoy, Pawan Goyal and Krithi Ramamritham, "Implications of Proxy Caching for Provisioning Servers and Networks," *Proceedings of ACM SIGMETRICS'2000 Conference, Santa Clara, CA*, pages 66-77, June 2000 *(acceptance rate: 17%)*.

C191. Venkata Duvvuri, Prashant Shenoy and Renu Tewari, "Adaptive Leases: A Strong Consistency Mechanism for the World Wide Web," *Proceedings of IEEE INFOCOM'2000, Tel Aviv, Israel*, pages 834-843, March 2000 *(acceptance rate: 27%)*.

C192. Jim Gray and Prashant Shenoy, "Rules of Thumb in Data Engineering", *Proceedings of the 16th IEEE International Conference on Data Engineering (ICDE'2000), San Diego CA*, pages 3-12, March 2000 *(invited paper)*.

C193. Prashant Shenoy, Pawan Goyal and Harrick M. Vin, "Architectural Considerations for Next Generation File Systems," *Proceedings of the Seventh ACM Conference on Multimedia, Orlando, FL*, pages 457-467, October 1999 *(acceptance rate: 18%)*.

C194. Sambit Sahu, Prashant Shenoy and Donald Towsley, "Design Considerations for Integrated Proxy Servers," *Proceedings of the Ninth IEEE International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'99), Basking Ridge, NJ*, pages 247–250, June 1999 *(acceptance rate: 36%)*.

C195. Prashant J. Shenoy and Harrick M. Vin, "Cello: A Disk Scheduling Framework for Next Generation Operating Systems," *Proceedings of the ACM SIGMETRICS Conference, Madison, WI*, pages 44–55, June 1998 *(acceptance rate: 17%)*.

C196. Prashant J. Shenoy, Pawan Goyal, Sriram S. Rao and Harrick M. Vin, "Symphony: An Integrated Multimedia File System," *Proceedings of the SPIE/ACM Conference on Multimedia Computing and Networking (MMCN'98), San Jose, CA*, pages 124–138, January 1998.

C197. Prashant J. Shenoy and Harrick M. Vin, "Efficient Striping Techniques for Multimedia File Servers," *Proceedings of the Seventh IEEE International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'97), St. Louis, MO*, pages 25–36, May 1997 *(acceptance rate: 25%)*.

C198. Banu Özden, Rajeev Rastogi, Prashant J. Shenoy, and Avi Silberschatz, "Fault-tolerant Architectures for Continuous Media Servers," *Proceedings of the ACM SIGMOD Conference, Montreal, Canada*, pages 79–90, June 1996 *(acceptance rate: 16%)*.

C199. Pawan Goyal, Harrick M. Vin, Chia Shen and Prashant J. Shenoy, "A Reliable, Adaptive Network Protocol for Video Transport," *Proceedings of IEEE INFOCOM'96, San Francisco, CA*, pages 1080–1091, March 1996 *(acceptance rate: 29.8%)*.

C200. Prashant J. Shenoy and Harrick M. Vin, "Efficient Support for Scan Operations in Video Servers," *Proceedings of the Third ACM Conference on Multimedia, San Francisco, CA*, pages 131–140, November 1995 *(acceptance rate: 22%)*

C201. Harrick M. Vin, Prashant J. Shenoy and Sriram Rao, "Efficient Failure Recovery in Multi-Disk Multimedia Servers," *Proceedings of the Twenty-Fifth Fault Tolerant Computing Symposium (FTCS), Pasadena, CA*, pages 12–21, June 1995 *(acceptance rate: 18.9%)*.

C202. Harrick M. Vin, Prashant J. Shenoy and Sriram Rao, "Analyzing the Performance of Asynchronous Disk Arrays for Multimedia Retrieval," *Proceedings of the First ISMM International Conference on Distributed Multimedia Systems and Applications, Honolulu, Hawaii*, Pages 14–17, August 1994.

Patents

P1. Pipelined data replication for disaster recovery, US Patent no 2014/0040206, Feb 6, 2014.

P2. System And Methods For Run Time Detection And Correction Of Memory Corruption, US Patent no 8,510,596, August 13, 2013

P3. Methods and Apparatus to Migrate Virtual Machines Between Distributive Computing Networks Across a Wide Area Network, US Pantent no 8,473,557, June 2013

P4. Fault tolerant Architectures for Continuous Media Servers US Patent no 6,079,028, June 2000.

Post-doctoral Fellows

1. Jeremy Gummeson, Research Fellow (2017-19, co-supervised with Deepak Ganesan), *Asst Professor, UMass Electrical and Computer Engineering*.
2. David Irwin, Post-doctoral Fellow (2007-12), *Asst Professor, UMass Electrical and Computer Engineering*.
3. Michael Zink, Post-doctoral Fellow (2004-08, co-supervised with Jim Kurose), *Asst Professor, UMass Electrical and Computer Engineering*.

Graduate Students

Ph.D Advisees:

1. Vani Gupta (Ph.D. Summer 2019), *Assistant Professor, Providence College* Thesis: Energy-aware Algorithms for Greening Internet-scale Distributed Systems Using Renewables.
2. Stephen Lee (Ph.D. Summer 2019), *Assistant Professor, University of Pittsburgh* Thesis: Software-defined Infrastructure for IoT-based Energy Systems,
3. Srinivasan Iyengar (Ph.D. Fall 2018), *Post-doc Researcher, Microsoft Research Bangalore* Thesis: Scalable data-driven modeling and analytics for smart buildings.
4. Prateek Sharma (Ph.D. Summer 2018), *Assistant Professor, Indiana University* Thesis: Transiency-Driven Resource Management For Cloud Computing Platforms.
5. Tian Guo (Ph.D., Summer 2016), *Assistant Professor, WPI*
   Thesis title: Elastic Resource Management in Distributed Clouds.

6. Xin He (Ph.D., Summer 2016), *Technical Staff Member, Oracle*
   Thesis title: Application Aware Resource Management for Cloud Platforms.

7. Vimal Mathew (Ph.D., Summer 2016), *Independent*
   Thesis title: Energy-efficient Content Delivery Networks.

8. Aditya Mishra (Ph.D, Summer 2015), *Assistant Professor, Seattle U*
   Thesis title: Energy Optimizations for Smart Buildings and Smart Grids.

9. Sean Barker (Ph.D, Summer 2014), *Assistant Professor, Bowdoin College*
   Thesis title: Model-driven Analytics of Energy Meter Data in Smart Homes.

10. Navin Sharma (Ph.D, Spring 2013), *Cofounder and CEO, Heystack Inc*
    Thesis title: Designing Distributed Systems for Intermittent Power

11. Rahul Singh (Ph.D, Spring 2013), *Research Staff Member, IBM T. J. Watson Research Center.*
    Thesis title: Resource Management for Enterprise Data center Applications.

12. Upendra Sharma (Ph.D, Spring 2013), *Research Staff Member, IBM T. J. Watson Research Center.*
    Thesis Title: Elastic Resource Management in Cloud Computing Platforms.

13. Timothy Wood (Ph.D, Fall 2011), *Assistant Professor, George Washington University.*
    Thesis Title: Improving Data Center Resource Management, Deployment and Availability With Virtualization.

14. Peter Desnoyers (Ph.D., Fall 2007), *Assistant Professor, Northeastern University.*
    Thesis Title: Distributed Sensing Networks: Archiving, Indexing, and Analysis

15. Purushottam Kulkarni (Ph.D., Fall 2006; co-advised) *Assistant Professor, Indian Institute of Technology Bombay, India*
    Thesis title: Design Issues in Multi-tier Heterogeneous Sensor Networks

16. Huan Li, (Ph.D., Summer 2006; co-advised), *Associate Professor, Beihang University , Beijing China*
    Thesis title: Resource Management for Distributed Real-Time Sensor Systems

17. Xiaotao Liu (Ph.D., Summer 2006; co-advised), *Senior Research Scientist, EMC*
    Thesis title: System Support for Pervasive Multimedia Systems

18. Vijay Sundaram (Ph.D., Fall 2005), *Software Engineer, Windows Division, Microsoft*
    Thesis: Self-managing Techniques for Storage Resource Management

19. Bhuvan Urgaonkar (Ph.D., Summer 2005) *Assistant Professor, Pennsylvania State University*
    Thesis title: Dynamic Resource Management in Internet Hosting Platforms

20. Abhishek Chandra (Ph.D.,Fall 2004) *Assistant Professor, University of Minnesota*,
    Thesis title: Resource Allocation for Self-Managing Servers

M.S Advisees:

1. Assan Toleuov (MS, Spring 2019)

2. Rishikesh Jha (MS, Spring 2019)

3. Ritesh Kumar (MS, Spring 2019)
4. Ishita Dasgupta (MS, Spring 2017),
5. Roopa Shenoy (MS, Spring 2017), Employer: Cisco
6. Puja Mishra, (MS, Summer 2016), Employer: Lutron
7. Nikita Mehra (MS, Summar 2016), Employer: Twitter
8. Seem Guggari (MS, Summer 2016), Employer: Microsoft
9. Anushree Ghosh (MS, Summer 2016), Employer: Microsoft
10. Sneha Narayan, (MS Summer 2015), Employer: Twilio
11. Sandeep Kalra (MS, Summer 2015) Employer: Amazon
12. Sippakorn Tansutthiwess, (M.S (ECE), Fall 2012), Employer: Amazon
13. Veena Udayabhanu, (M.S., Spring 2012), Employer: Microsoft
14. Aditya Nemmuluri, (M.S., Spring 2008; co-advised), Employer: Cisco Corporation
15. Rick Cocci, (M.S., Spring 2008; co-advised), Employer: State Street Corporation
16. Srinath Gaddam (M.S., Summer 2004), Employer: EMC
17. Peter Radkov (M.S., Summer 03), Employer: Sirma Corporation, Bulgaria
18. Jiang Lan (M.S., Summer 02), Employer: Verizon Corporation
19. Saif Hasan (M.S., Summer 01) Employer: Microsoft Corporation
20. Anoop Ninan(M.S., Summer 01) Employer: EMC Corporation
21. M S. Raunak (M.S., Summer00)
22. Venkata Duvvuri (M.S.,Summer 99) Inital Employer: Goldman Sachs

University and State-level Service

- *Associate Dean*, College of Information and Computer Sciences, 2016-present.
- Director, Center for Personalized Health Monitoring, 2018-19.
- Director, Center for Smart and Connected Society, 2017-present.
- Member, Information and Communication Technology Council, UMass Faculty Senate, 2015-present.
- Member, Campus Physical Planning Council, UMass Faculty Senate, 2017-present.
- Co-Chair, UMass Research IT Strategic Planning committee. 2014-15.
- Member, University CIO Search Committee, 2013
- Member, CVIP director search committee, 2012.
- *Chair, Informatics Program Committee* 2012-14 to create a new undergraduate degree program in Informatics.
- Member ECE Faculty Search committee, 2010-11
- Member of the Research and Education subgroup to set future directions for the Massachusetts Green High Performance Computing Center (MGHPCC), 2009-11. The MGHPCC is a collaboration between five Universities in partnership with the Commonwealth of Massachusetts, Cisco, and EMC to build a state of the art, environmentally responsible computing center.
- Worked to define the research vision for the MGHPCC; conducted workshops for Holyoke K-12 teachers and local community college faculty to incorporate sustainability and green computing concepts in the classroom; Working with HG&E to gather and analyze energy usage data from Holyoke residential and business consumers to enhance energy efficiency.
- Member of ad-hoc committee on " Moving CS to the College of Engineering", 2009-10.
- Member, CS Department Chair Search Committee, 2006-07.
- PEEAS/VIP Distance Education Advisory Committee, 2003-04 and 2004-05.

Departmental Service

Note: Chair responsibilities are italicized.

- *Chair, AFR Personnel Committee*, 2016-16 and 2016-17.
- *Chair, Space Task Force* 2015-2017
- *Chair, Strategic Planning Committee, 2008-09*, 2009-2010, *2010-2011*.
- *Chair, Computing Committee, 2008-09, 2009-10, 2010-11*
- Member, Executive Committee, 2008-09, 2010-11.
- Member, Personnel Committee, and *Co-Chair Annual Faculty Report (AFR) Evaluation Committee, 2007-08*.
- Member, Strategic Planning Committee, 2007-08 and 2006-07.

- Member, Personnel Committee, 2006-07.
- *Chair, Special Events Committee, 2006-07.*
- Member, Website Redesign Committee, 2006-08.
- Member, Space committee, Fall 2006.
- *Chair, Faculty Recruiting, 2004-05.*
- Member, Diversity Committee, 2004-05.
- Member, Library Committee, 2004-05 and 2003-04.
- *Chair, Ad-hoc Committee on Strategic Planning, 2003-04.*
- Member, Computing Committee, 2003-04, 2015-2017.
- Member, Faculty Recruiting Committee, 2001-02 and 2002-03.
- *Chair, Computing Committee, 2000-01 and 2001-02.*
- Member, Personal Committee, 2000-01.
- Member, Executive Committee, 1999-2000.
- Member, Computing Committee, 1999-2000.
- Member, Graduate Admissions Committee, 1998-99.

National, International and Professional Service

- **National-level and International Service**
  - Cofounder and Chair, ACM Emerging Interest Group in Energy.
  - Co-organiser of a NSF workshop on Energy Efficiency in Data Centers.
  - Co-organiser of a CRA's CCC workshop to develop a vision and the research agenda for the "Role of Information Technology in Sustainability." 2011.
  - Member of an ad-hoc IUCEE committee to improve the quality of Computer Science Ph.D programs in emerging economies such as India (2010-11).
  - Organizer of a week-long IUCEE workshop to enhance graduate-level curriculum in Computer Science, and in the area of Distributed Computing Systems, New Delhi, India, 2010.
- **Steering Committees**
  - IEEE Trans. on Mobile Computing Steering Committee (2014-2016)
  - ACM e-Energy Conference Steering Committee (2014-2017, chair)
  - ACM BuildSys Conference Steering Committee (2014-2016)
- **Journal Editorial Boards** and **Guest Editor for Special Issues**
  - Editorial board member,*ACM Transaction on Internet of Things (TIOT)*, 2018-present.

- Editorial board member,*ACM Transaction on Modeling and Performance Evaluation of Computing Systems*, 2015-2016.
- Editorial board member, *ACM Transaction on the Web (TWEB)*, 2008-2016.
- Editorial board member, *Sustainable Energy, Grids and Networks Journal*, 2014-2016.
- Editorial board member, *Springer/ACM Multimedia Systems Journal*, 2006-present.
- Editorial board member, *PeerJ Computer Science*, 2014-present.
- Guest Editor, *IEEE Internet Computing*, Special Issues on Internet Data Dissemination 2007.
- Guest Editor, *Multimedia Tools and Applications*, 2005.
- Guest Editor, *ACM Transcaction on Multimedia Communication and Applications (TOMCAP)*, February 2005.
- Guest Editor, *IEEE Communications Maganize*, Special Issue on Proxy Services for the Internet, August 2004.
- Guest Editor, *ACM/Springer Multimedia Systems Journal*, Special Issue on Multimedia Computing and Networking, 2003.

- **Program Chair, General Chair and Other conference organization:**
  - General Chair, *ACM/IEEE IotDI 2020*
  - Program Chair, *ACM/IEEE IotDI 2019*
  - Program Chair, *ACM APSys 2017*
  - Program Chair, *IEEE Intl Conference on Network Protocols 2016*
  - Program Chair, *IEEE Cloud and Autonomic Computing 2015*
  - Program Chair, *ACM eEnergy 2014*
  - Program Chair, *ACM Buildsys 2013*.
  - Program Chair, *IEEE COMSNETS 2013* conference.
  - Co-founded the HotCloud Workshop series in the Cloud Computing Area. Program Chair, *USENIX HotCloud* 2009;
  - Program Chair, *ACM Sigmetrics* 2008
  - Program Chair, *Performance 2007* conference.
  - Program Chair, *World Wide Web 2007* conference.
  - Program Chair, *Multimedia Information Systems* Workshop, College Park, MD, August 2004.
  - Program Chair, *ACM Multimedia Conference*, Berkeley CA, November 2003.
  - Program Chair, *ACM SIGMultimedia/SPIE Multimedia Computing and Networking (MMCN) Conferer-ence*, January 2002.
  - Program Vice Chair, *World Wide Web Conference*, Japan 2005.
  - Deputy Chair, *World Wide Web Conference*, New York, 2004.
  - General Chair, *ACM SIGMultimedia/SPIE Multimedia Computing and Networking (MMCN) Confer-ence*, January 2003.
  - Associate Chair, *ACM Multimedia Conference*, December 2002.

  - Student Posters Chair, *ACM SIGCOMM 2004*.
  - Tutorials Chair, *ACM SIGMETRICS* 2002.

- **Program Committee and Referee activities**

  - Program Committee Member, *ACM MMSys 2013, ICDCS 2103, NOSSDAV 2012, ACM CoNext 2102, MMSys 2012, E6 2012, GreenNEts 2012, ACM Buildsys 2012, SOCC 2011, Sigmetrics 2011, MMSys 2011, Sensys 2010, USENIX 2009, ACM Sigmetrics 2009, COMSNET 2009, MMCN 2009, ICNP 2008, ACM Multimedia 2008, ACM NOSSDAV 2008, IEEE LANMAN 2008, ACM HotMetrics 2008, ACM Multimedia 2007, ACM NOSSDAV 2007, ACM Multimedia 2006, ACM NOSSDAV 2006, IEEE Infocom 2006, IEEE Infocom 2005, ACM Multimedia 2005, ACM Nossdav 2005, COMAD 2005, Sigmetrics 2004, Global Internet 2004, ACM MDM 2004, MMCN 2004, IEEE WIAPP 2003, IEEE ICDE 2003, IEEE RTSS 2002, ACM NOSSDAV 2002, WWW 2002, Packet Video Workshop 2002, ACM Multimedia 2001, ACM/SPIE MMCN'2001, NOSSDAV'2000, ACM SIGMETRICS 2000, ACM SIGMETRICS Workshop on Internet Server Performance (WISP'99), IEEE ICDCS'99*.
  - Referee for numerous journals: *ACM TOCS, ACM TIOT, IEEE TOC, IEEE TSE, IEEE TPDS, IEEE TKDE, IEEE/ACM TON, IEEE TMM, Journal of Parallel and Distributed Computing, ACM/Springer Multimedia Systems Journal, Kluwer Telecommunication Systems Journal, IEEE Multimedia Magazine, and IEEE Internet Magazine*.

- Panelist for various National Science Foundation (NSF) panels

Federal and State Funding for Research and Education

G1. DOD Army Research Lab: Internet of Battlefield Things Collaborative Research Alliance (UMass PI and Area Lead), 2017-2022, $4.5M for the first five 5 years.

G2. NSF SCH grant: Enhancing Context-Awareness and Personalization for Intensively Adaptive Smoking Cessation Messaging Interventions (co-PI), $1M, 2017-20.

G3. NSF CPS grant: Software Defined Solar Systems, $500K 2017-20

G4. NASA Contract: Integrating Inference and Complex Event Processing for RFID Logistics Management (PI), $1.2M, 2014-2019

G5. DOD Army Research Lab, Improved Systems for Real- World Pervasive Human Sensing (co-PI), 2015-2016.

G6. NSF PFI:BIC grant- Utility-driven Smart Energy Services (PI), $1M, 9/2015-8/2018.

G7. EPA grant: WINNS: Water Innovation Network for Sustainable Small Systems (UMass co-PI), 2015-2018.

G8. NSF: CPS-Security: Enhancing Privacy in Smart Buildings and Homes (co-PI), $500K, 2015-2018

G9. NSF CSR grant: System Support for Transiency in Data Center and Cloud Computing (PI) $500K, 2015-18

G10. NSF Computing Research Infrastructure Award (PI), A Programmable Data-Driven Testbed for Sustainable Buildings Research, $587K, 2014-17.

G11. NSF FIA-NP: Collaborative Research: The Next-Phase MobilityFirst Project - From Architecture and Protocol Design to Advanced Services and Trial Deployments, $1.3M, 2013-15.

G12. MA DOER grant, UMASS Amherst Energy Extension Initiative (PI), $6M, 2014-2017.

G13. Mass Life Science Center award: MHGPCC Life science cluster, $4.5M, July 2013-16.

G14. NSF MRI award for the MGHPCC, $2.3M, October 2012-2015

G15. National Science Foundation OCI-1032765 (UMass PI), "Collaborative Research: SDCI The Missing Link: Connecting Eucalyptus Clouds with Multi-Layer Networks", (UMass share of award $115K), Sept 2010-Aug 2013.

G16. National Science Foundation Computer Systems Research Award CNS-0916972 (Co-PI), " High Assurance at Low Cost in Data Centers Using Virtualization," $500K, (Sept 2009-Aug 2012).

G17. NSF CNS-0916577 Computer Systems Research Award (Co-PI) "System and Network Support for Harvesting-aware Solar Sensor Networks," $500K, (Sept 2009- Aug 2012).

G18. NSF Global Environment for Network Innovation (GENI) Grant (Co-PI), "DiCloud: A Data-intensive Cloud Testbed," $500K, (Duration Oct 2009 - Sept 2012). All GENI grants are administered by BBN Technologies via subcontract.

G19. National Science Foundation Computing Research Infrastructure award CNS-0855128 "Design of a Solar-powered River Sensor Network Tested," $662K, (Sept 2009-Aug 2012).

G20. UMass President's Science and Technology Award (Co-PI), "The Massachusetts Green High Performance Computing Center," $200K, Sept 2009-Aug 2011.

G21. NSF Global Environment for Network Innovation (GENI) Grant (PI), "Sensor Virtualization and Slivering in an Outdoor Wide-area Wireless Sensor/Actuator Network Testbed," $490K, (Duration Oct 2008 - Sept 2011). All GENI grants are administered by BBN Technologies via subcontract.

G22. UMass President's Science and Technology Grant, "The Massachusetts Center for Water and Environmental Sensing Technologies (MassWEST)," Co-directors: B. Levine and P. Shenoy, $175K, Sept 2007.

G23. NSF Computer Systems Research Award CNS-0720616 (PI), "Measurement, modeling and Analysis of Large Complex Data Centers", $200K, (Duration 9/07-8/10).

G24. NSF Computer Systems Research Award CNS-0720271 (PI), "Dynamic recofiguration in Virtualized Data Centers", $330K, (Duration 9/07-8/09).

G25. NSF Embedded Systems Research Award CNS-0615075 (Co-PI), "ASPIRE: Antipodal Staged Processing for Role-adaptive Embedded Systems", Joint grant with UCLA and Yale, $650K, (Duration: Sept 2006 – Aug 2009).

G26. NSF Network of Sensor Systems (NOSS) Award CNS-0626873 (Co-PI), "STONES: Storage-centric Network Embedded Systems," $500,000 (Duration: Sept 2006 – Aug 2009).

G27. NSF Major Research Instrumentation Award (MRI), CNS-052072 (PI), "Acquisition of a Laboratory Testbed for Networked Embedded Systems and Sensor Research," $300,000, (Sept 2005-August 2009). Joint with Amherst College and Mount Holyoke; submitted through Five Colleges, Inc.

G28. NSF Information Technology Research Award ANI-0325868 (Co-PI), "Hyperion - Next Generation Measurement Infrastructure and Application Use," $2,700,000, September 2003 (Duration: 9/03-8/09).

G29. NSF Engineering Research Center (ERC), (Affiliated Faculty), "Center For Collaborative Adaptive Sensing of the Atmosphere (CASA)," $17,000,000 (and $13,428,929 in cost-sharing), September 2003 (Duration: 9/03-8/09, with a planned extension through 2013).

G30. NSF Research Resources Award EIA-0323597, (PI), "Instrumenting and Measuring Distributed Internet Applications," $444,000 (including cost-sharing), September 2003 (Duration: 9/03-8/06).

G31. NSF Information Technology Research Award, CCR-0219520 (PI), "System Support for Mobile Multimedia and Web Applications," $350,000, September 2002 (Duration: 9/02-8/05).

G32. NSF Distributed Systems Program Award CCR-0098060 (PI), "Proxy-based Dissemination of Dynamic Web Data," $140,000, April 2001 (Duration: 9/01-8/04).

G33. NSF Combined Research and Curriculum Development (CRCD) Award EIA-0087945 (Co-PI), "Curriculum Development and Infrastructure for an Advanced Systems Laboratory," $496,000, September 2000 (Duration: 9/00-8/04).

G34. NSF Research Infrastructure Award EIA-0080119 (Co-PI), "Infrastructure to Support Mixed Wired/Wireless Information Systems," $992,585, September 2000 (Duration: 9/00-8/05).

G35. NSF Career Award CCR-9984030 (PI), "Scalable High-Performance Information Servers for Internet-based Multimedia Applications," $200,000, April 2000 (Duration: 4/00-3/04).

G36. NSF Special Projects in Networking Award ANI 9977635 (Co-PI), "Proxy Services in Wide Area Networks," $1,010,263, September 1999 (Duration: 9/99-8/03).

Industrial Funding for Research and Education

G37. Holyoke Gas and Electric award, $60,000 2019

G38. Veea Labs, gift, $20,000, 2018.

G39. Uptycs gift, $15,000, 2018

G40. Cisco Research award, 2015.

G41. Google Faculty Research Award, 2015.

G42. Embue Unrestricted Grant, 2015.

G43. NEC Labs Gift (PI), "Cloud Bursting and Disaster Recovery for Cloud-based Database Applications", $60,000, Jan 2011.

G44. IBM Gift (PI), "Research on Cloud-based Smarter Planet Services," $50K, 2011

G45. Google Gift (Co-PI), "Scalable One Pass Analytics for Stream Processing", $50K, 2010.

G46. IBM Faculty Award (PI), "Research on Cloud Computing," 2009.

G47. Intel Corporation Gift (PI), "Resource Management in Virtualized Data Centers ," $180,000. November 2005, 2006 and 2007.

G48. AT&T VURI Award (PI), "Resource management in Data Centers," $35,000. Sept 2007.

G49. Intel Corporation, Research Gift (Co-PI), "An IXP-Based Measurement Node for Data-Center Traffic Analysis," $75,000, July 2003.

G50. Microsoft Corporation, Research Gift (PI), "Consistent Data Dissemination and Updates in PDA-based Adhoc Networks," $15,000, August 2003.

G51. IBM University Partnership Award (PI), "Design of Large Self-Managing Storage Systems," $30,000, July 2003.

G52. Microsoft Corporation, Hardware and Software Donation (PI), "Course on .NET Technologies," $40,000, June 2003.

G53. Microsoft Corporation, Hardware and Software donation of 35 workstations and a server (PI), "A Teaching Laboratory for Multimedia and Networking Courses," $75,000, June 2002.

G54. IBM Faculty Partnership Award (PI), " Proxy Services for the Internet," $40,000, May 2001.

G55. Sprint Corporation, Research Gift (PI), "Operating Systems Support for Server Clusters," $40,000, May 2001.

G56. Intel Corporation, Equipment Donation (PI), "A Teaching Laboratory for Systems Courses," $30,000, May 2001.

G57. Microsoft Corporation, Equipment Donation (PI), "Research in Proxy-based Mobile Information Access," $10,000 (approximate value), February 2001.

G58. EMC Corporation, Research Gift (PI), "Design of a Storage and Delivery Architecture for Multimedia Data," $50,375, September 2000.

G59. CAIDA Internet Teaching Laboratory (ITL) Award (PI), "An Educational Laboratory for the Study of Multimedia, Networks, and Security," September 2000.

G60. Sprint Corporation, Research Gift (PI), "OS Resources Management for Server Clusters," $20,000, June 2000.

G61. IBM Faculty Partnership Award (PI), "Web Proxy Services for Wide Area Networks," $30,000, May 2000.

G62. Intel equipment donation (PI), "Infrastructure for a Multimedia Teaching Laboratory", $15,000, April 2000.

G63. Intel Corporation, Research Gift (Co-PI), "Measurement, Adaptation and Coordinated Resource Use in Distributed Multimedia Applications," $120,000, July 1999.

G64. Intel Corporation, Equipment donation (Co-PI), $16,000 (approximate value), December 1999.

G65. Microsoft Corporation, Software donation (PI), $2000 (approximate value), March 1999.

APPENDIX 2: MATERIALS CONSIDERED IN THE PREPARATION OF THIS DECLARATION

Exhibit No.	Description
1001	U.S. Patent No. 8,572,138
1002	File History for U.S. Patent No. 8,572,138
1004	U.S. Patent No. 7,577,722 to Khandekar *et al*. ("Khandekar")
1005	U.S. Patent No. 8,209,680 to Le *et al*. ("Le")
1006	U.S. Patent No. 7,962,633 to Sidebottom *et al*. ("Sidebottom")
1007	U.S. Patent Application Publication No. 2003/0036886 to Stone ("Stone")
1008	Khanna *et al*., *Application Performance Management in Virtualized Server Environments*, NOMS 2006 ("Khanna")
1009	Urgaonkar *et al*., *Dynamic Provisioning of Multi-tier Internet Applications*, ICAC'05 ("Urgaonkar")
1010	Excerpt of S. Russell, P. Norvig, *Artificial Intelligence - A Modern Approach* (2d. ed. Prentice-Hall 2003) ("Russell")
1011	Barham *et al*., *Xen and the Art of Virtualization*, SOSP '03: ACM Symposium on Operating Systems Principles, 164-177 (Oct., 2003)
1012	U.S. Patent No. 6,496,847 to Bugnion *et al*. ("Bugnion")
1013	U.S. Patent Application Publication No. 2004/0019672 to Das *et al*. ("Das")
1014	U.S. Patent Application Publication No. 2007/0220246 to Powell *et al*. ("Powell")
1015	JP Patent Application Publication No. 2001-318797 to French
1016	English translation of JP Patent Application Publication No. 2001-318797 to French ("French")

APPENDIX 3: CHALLENGED CLAIMS

1[a]. A distributed computing system comprising:

[1b] a software image repository comprising non-transitory, computer-readable media operable to store:

[1c] (i) a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes,

[1d] wherein when executed on the applications nodes, the image instances of the virtual machine manager provide a plurality of virtual machines, each of the plurality of virtual machine operable to provide an environment that emulates a computer platform, and

[1e] (ii) a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines; and

[1f] a control node that comprises an automation infrastructure to provide autonomic deployment of the plurality of image instances of the virtual machine manager on the application nodes by causing the plurality of image instances of the virtual machine manager to be copied from the software image repository to the application nodes and

[1g] to provide autonomic deployment of the plurality of image instances of the software applications on the virtual machines by causing the plurality of

image instances of the software applications to be copied from the software image repository to the application nodes.

2. The distributed computing system of claim 1, wherein the image instances of the plurality of software applications comprise virtual disk image files.

3. The distributed computing system of claim 1, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises an operating system.

4. The distributed computing system of claim 1, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises a server application.

5. The distributed computing system of claim 1, wherein a first one of the virtual machines provides an environment that emulates an x86 platform.

9. The distributed computing system of claim 1, wherein the control node further comprises one or more rule engines that provide autonomic deployment of the software applications to the virtual machines in accordance with a set of one or more rules.

10. The distributed computing system of claim 9, wherein the one or more rule engines are forward-chaining rule engines.

17. The distributed computing system of claim 1, wherein the application nodes are organized into tiers, and wherein status data is collected based on the tiers.

19[a]. A method comprising:

[19b] storing a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes in a distributed computing system,

[19c] wherein the image instances of the virtual machine manager provide a plurality of virtual machines, each virtual machine operable to provide an environment that emulates a computer platform;

[19d] storing a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines;

[19e] executing a rules engine to perform operations to autonomically deploy the image instances of the virtual machine manager on the application nodes by causing the image instances of the virtual machine manager to be copied to the application nodes; and

[19f] executing the rules engine to perform operations to autonomically deploy the image instances of the software applications on the virtual machines by causing the image instances of the software applications to be copied to the application nodes.

20. The method of claim 19, wherein storing the image instances of the plurality of software applications comprises storing virtual disk image files.

21. The method of claim 19, wherein a first image instance of the plurality of image instances of the plurality of software applications comprises an operating system.

26[a]. A non-transitory, computer-readable medium comprising instructions stored thereon, that when executed by a programmable processor:

[26b] store a plurality of image instances of a virtual machine manager that is executable on a plurality of application nodes in a distributed computing system,

[26c] wherein the image instances of the virtual machine manager provide a plurality of virtual machines, each virtual machine operable to provide an environment that emulates a computer platform;

[26d] store a plurality of image instances of a plurality of software applications that are executable on the plurality of virtual machines;

[26e] perform operations to autonomically deploy the image instances of the virtual machine manager on the application nodes by causing the image instances of the virtual machine manager to be copied to the application nodes; and

[26f] perform operations to autonomically deploy the image instances of the software applications on the virtual machines by causing the image instances of the software applications to be copied to the application nodes.

APPENDIX 4: UNDERSTANDING OF THE LAW

I have applied the following legal principles provided to me by counsel in arriving at the opinions set forth in this report.

Legal Standard for Prior Art

I am not an attorney. I have been informed by attorneys of the relevant legal principles and have applied them to arrive at the opinions set forth in this declaration.

I understand that the petitioner for inter partes review may request the cancelation of one or more claims of a patent based on grounds available under 35 U.S.C. § 102 and 35 U.S.C. § 103 using prior art that consists of patents and printed publications.

Anticipation and Prior Art

I understand that § 102 specifies when a challenged claim is invalid for lacking novelty over the prior art, and that this concept is also known as "anticipation." I understand that a prior art reference anticipates a challenged claim, and thus renders it invalid by anticipation, if all elements of the challenged claim are disclosed in the prior art reference. I understand the disclosure in the prior art reference can be either explicit or inherent, meaning it is necessarily present or implied. I understand that the prior art reference does not have to use the same words as the challenged claim, but all of the requirements of the claim must

be disclosed so that a person of ordinary skill in the art could make and use the claimed subject-matter.

In addition, I understand that § 102 also defines what is available for use as a prior art reference to a challenged claim.

Under § 102(a), a challenged claim is anticipated if it was patented or described in a printed publication in the United States or a foreign country before the challenged claim's date of invention.

Under § 102(b), a challenged claim is anticipated if it was patented or described in a printed publication in the United States or a foreign country more than one year prior to the challenged patent's filing date.

Under § 102(e), a challenged claim is anticipated if it was described in a published patent application that was filed by another in the United States before the challenged claim's date of invention, or was described in a patent granted to another that was filed in the United States before the challenged claim's date of invention.

I understand that a challenged claim's date of invention is presumed to be the challenged patent's filing date. I also understand that the patent owner may establish an earlier invention date and "swear behind" prior art defined by § 102(a) or § 102(e) by proving (with corroborated evidence) the actual date on which the

named inventors conceived of the subject matter of the challenged claim and proving that the inventors were diligent in reducing the subject matter to practice.

I understand that the filing date of a patent is generally the filing date of the application filed in the United States that issued as the patent. However, I understand that a patent may be granted an earlier effective filing date if the patent owner properly claimed priority to an earlier patent application.

I understand that when a challenged claim covers several structures, either generically or as alternatives, the claim is deemed anticipated if any of the structures within the scope of the claim is found in the prior art reference.

I understand that when a challenged claim requires selection of an element from a list of alternatives, the prior art teaches the element if one of the alternatives is taught by the prior art.

Legal Standard for Obviousness

I understand that even if a challenged claim is not anticipated, it is still invalid if the differences between the claimed subject matter and the prior art are such that the claimed subject matter would have been obvious to a person of ordinary skill in the pertinent art at the time the alleged invention.

I understand that obviousness must be determined with respect to the challenged claim as a whole.

I understand that one cannot rely on hindsight in deciding whether a claim is obvious.

I also understand that an obviousness analysis includes the consideration of factors such as (1) the scope and content of the prior art, (2) the differences between the prior art and the challenged claim, (3) the level of ordinary skill in the pertinent art, and (4) "secondary" or "objective" evidence of non-obviousness.

Secondary or objective evidence of non-obviousness includes evidence of: (1) a long felt but unmet need in the prior art that was satisfied by the claimed invention; (2) commercial success or the lack of commercial success of the claimed invention; (3) unexpected results achieved by the claimed invention; (4) praise of the claimed invention by others skilled in the art; (5) taking of licenses under the patent by others; (6) deliberate copying of the claimed invention; and (7) contemporaneous and independent invention by others. However, I understand that there must be a relationship between any secondary evidence of non-obviousness and the claimed invention.

I understand that a challenged claim can be invalid for obviousness over a combination of prior art references if a reason existed (at the time of the alleged invention) that would have prompted a person of ordinary skill in the art to combine elements of the prior art in the manner required by the challenged claim. I understand that this requirement is also referred to as a "motivation to combine,"

"suggestion to combine," or "reason to combine," and that there are several rationales that meet this requirement.

I understand that the prior art references themselves may provide a motivation to combine, but other times simple common sense can link two or more prior art references. I further understand that obviousness analysis recognizes that market demand, rather than scientific literature, often drives innovation, and that a motivation to combine references may come from market forces.

I understand obviousness to include, for instance, scenarios where known techniques are simply applied to other devices, systems, or processes to improve them in an expected or known way. I also understand that practical and common-sense considerations should be applied in a proper obviousness analysis. For instance, familiar items may have obvious uses beyond their primary purposes.

I understand that the combination of familiar elements according to known methods is obvious when it yields predictable results. For instance, obviousness bars patentability of a predictable variation of a technique even if the technique originated in another field of endeavor. This is because design incentives and other market forces can prompt variations of it, and predictable variations are not the product of innovation, but rather ordinary skill and common sense.

I understand that a particular combination may be obvious if it was obvious to try the combination. For example, 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. This would result in something obvious because the result is the product not of innovation but of ordinary skill and common sense. However, I understand that it may not be obvious to try a combination when it involves unpredictable technologies.

It is further my understanding that a proper obviousness analysis focuses on what was known or obvious to a person of ordinary skill in the art, not just the patentee. Accordingly, I understand that any need or problem known in the field of endeavor at the time of invention and addressed by the patent can provide a reason for combining the elements in the manner claimed.

It is my understanding that the Manual of Patent Examining Procedure §2143 sets forth the following as exemplary rationales that support a conclusion of obviousness:

- Combining prior art elements according to known methods to yield predictable results;
- Simple substitution of one known element for another to obtain predictable results;
- Use of known technique to improve similar devices (methods, or products) in the same way;

- Applying a known technique to a known device (method, or product) ready for improvement to yield predictable results;
- Choosing from a finite number of identified, predictable solutions, with a reasonable expectation of success;
- Known work in one field of endeavor may prompt variations of it for use in either the same field or a different one based on design incentives or other market forces if the variations are predictable to one of ordinary skill in the art;
- Some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

A person of ordinary skill in the art looking to overcome a problem will often use the teachings of multiple publications together like pieces of a puzzle, even though the prior art does not necessarily fit perfectly together. Therefore, I understand that references for obviousness need not fit perfectly together like puzzle pieces. Instead, I understand that obviousness analysis takes into account inferences, creative steps, common sense, and practical logic and applications that a person of ordinary skill in the art would employ under the circumstances.

I understand that a claim can be obvious in light of a single reference, if the elements of the challenged claim that are not explicitly or inherently disclosed in the reference can be supplied by the common sense of one of skill in the art.

I understand that obviousness also bars the patentability of applying known or obvious design choices to the prior art. One cannot patent merely substituting one prior art element for another if the substitution can be made with predictable results. Likewise, combining prior art techniques that are interoperable with respect to one another is generally obvious and not patentable.

In order for a claim to be found invalid based upon a modification or combination of the prior art, there must be reasonable expectation that a person of ordinary skill would have successfully modified or combined the prior art to arrive at the claimed arrangement. This does not mean that it must be certain that a person of ordinary skill would have been successful – the law only requires that the person of ordinary skill in the art would have perceived a reasonable expectation of success in modifying or combining the prior art to arrive at the claimed invention.

In sum, my understanding is that obviousness invalidates claims that merely recite combinations of, or obvious variations of, prior art teachings using understanding and knowledge of one of skill in the art at the time and motivated by the general problem facing the inventor at the time. Under this analysis, the prior art references themselves, or any need or problem known in the field of endeavor

at the time of the invention, can provide a reason for combining the elements of or attempting obvious variations on prior art references in the claimed manner.

Legal Standard for Claim Construction

I understand that before any invalidity analysis can be properly performed, the scope and meaning of the challenged claims must be determined by claim construction.

I understand that a patent may include two types of claims, independent claims and dependent claims. I understand that an independent claim stands alone and includes only the limitations it recites. I understand that a dependent claim depends from an independent claim or another dependent claim. I understand that a dependent claim includes all the limitations that it recites in addition to the limitations recited in the claim (or claims) from which it depends.

In comparing the challenged claims to the prior art, I have carefully considered the patent and its file history in light of the understanding of a person of skill at the time of the alleged invention.

I understand that to determine how a person of ordinary skill would have understood a claim term, one should look to sources available at the time of the alleged invention that show what a person of skill in the art would have understood disputed claim language to mean. It is my understanding that this may include what is called “intrinsic” evidence as well as “extrinsic” evidence.

I understand that, in construing a claim term, one should primarily rely on intrinsic patent evidence, which includes the words of the claims themselves, the remainder of the patent specification, and the prosecution history. I understand that extrinsic evidence, which is evidence external to the patent and the prosecution history, may also be useful in interpreting patent claims when the intrinsic evidence itself is insufficient. I understand that extrinsic evidence may include principles, concepts, terms, and other resources available to those of skill in the art at the time of the invention.

I understand that words or terms should be given their ordinary and accepted meaning unless it appears that the inventors were using them to mean something else or something more specific. I understand that to determine whether a term has special meaning, the claims, the patent specification, and the prosecution history are particularly important, and may show that the inventor gave a term a particular definition or intentionally disclaimed, disavowed, or surrendered claim scope.

I understand that the claims of a patent define the scope of the rights conferred by the patent. I understand that because the claims point out and distinctly claim the subject matter which the inventors regard as their invention, claim construction analysis must begin with and is focused on the claim language itself. I understand that the context of the term within the claim as well as other claims of the patent can inform the meaning of a claim term. For example, because

claim terms are normally used consistently throughout the patent, how a term is used in one claim can often inform the meaning of the same term in other claims. Differences among claims or claim terms can also be a useful guide in understanding the meaning of particular claim terms.

I understand that a claim term should be construed not only in the context of the particular claim in which the disputed term appears, but in the context of the entire patent, including the entire specification. I understand that because the specification is a primary basis for construing the claims, a correct construction must align with the specification.

I understand that the prosecution history of the patent as well as art incorporated by reference or otherwise cited during the prosecution history are also highly relevant in construing claim terms. For instance, art cited by or incorporated by reference may indicate how the inventor and others of skill in the art at the time of the invention understood certain terms and concepts. Additionally, the prosecution history may show that the inventors disclaimed or disavowed claim scope, or further explained the meaning of a claim term.

With regard to extrinsic evidence, I understand that all evidence external to the patent and prosecution history, including expert and inventor testimony, dictionaries, and learned treatises, can also be considered. For example, technical dictionaries may indicate how one of skill in the art used or understood the claim

terms. However, I understand that extrinsic evidence is considered to be less reliable than intrinsic evidence, and for that reason is generally given less weight than intrinsic evidence.

I understand that in general, a term or phrase found in the introductory words or preamble of the claim, should be construed as a limitation if it recites essential structure or steps, or is necessary to give meaning to the claim. For instance, I understand preamble language may limit claim scope: (i) if dependence on a preamble phrase for antecedent basis indicates a reliance on both the preamble and claim body to define the claimed invention; (ii) if reference to the preamble is necessary to understand limitations or terms in the claim body; or (iii) if the preamble recites additional structure or steps that the specification identifies as important.

On the other hand, I understand that a preamble term or phrase is not limiting where a challenged claim defines a structurally complete invention in the claim body and uses the preamble only to state a purpose or intended use for the invention. I understand that to make this determination, one should review the entire patent to gain an understanding of what the inventors claim they invented and intended to encompass in the claims.

I understand that 35 U.S.C. § 112 ¶ 6 created an exception to the general rule of claim construction called a "means plus function" limitation. These types of

terms and limitations should be interpreted to cover only the corresponding structure described in the specification, and equivalents thereof. I also understand that a limitation is presumed to be a means plus function limitation if (a) the claim limitation uses the phrase "means for"; (b) the "means for" is modified by functional language; and (c) the phrase "means for" is not modified by sufficient structure for achieving the specified function.

I understand that a structure is considered structurally equivalent to the corresponding structure identified in the specification only if the differences between them are insubstantial. For instance, if the structure performs the same function in substantially the same way to achieve substantially the same result. I further understand that a structural equivalent must have been available at the time of the issuance of the claim.