

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent of: Vipul Pabari
U.S. Patent No.: 11,656,915 Attorney Docket No.: 42342-0156IP1
Issue Date: May 23, 2023
Appl. Serial No.: 17/074,500
Filing Date: October 19, 2020
Title: VIRTUAL SYSTEMS MANAGEMENT

Mail Stop Patent Board

Patent Trial and Appeal Board
U.S. Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

**PETITION FOR *INTER PARTES* REVIEW OF UNITED STATES PATENT
NO. 11,656,915 PURSUANT TO 35 U.S.C. §§ 311–319, 37 C.F.R. § 42**

TABLE OF CONTENTS

I.	REQUIREMENTS	1
A.	Standing	1
B.	IPR Challenge	1
C.	Claim Construction	3
1.	“control computer”	4
2.	“virtual assets”	4
3.	“resources”/“physical resources”	5
4.	“network resource”	5
5.	“software instantiations of computer systems”	5
6.	“provisioning”/“provisioned”	6
7.	“control computer connection”	6
8.	“logical”	7
9.	“physical storage drives”	8
D.	Level of Ordinary Skill in the Art	8
II.	THE ’915 PATENT	9
A.	Specification	9
B.	Relevant Prosecution	9
III.	UNPATENTABILITY GROUNDS	10
A.	GROUNDS 1A/1B	10
1.	Khandekar	10
2.	Analysis	14
B.	GROUND 1C	26
1.	Khandekar-Bandhole	26
2.	Analysis	28
C.	GROUND 1D	31
1.	Khandekar-Vasilevsky	31
2.	Analysis	36
D.	GROUND 1E	43
1.	Khandekar-Fujino	43
2.	Analysis	46
E.	GROUND 2A	48
1.	Hirschfeld-Smith	48
2.	Analysis	57
F.	GROUND 2B	70
1.	Hirschfeld-Smith-Bandhole	70
2.	Analysis	71

G.	GROUND 2C.....	75
1.	Hirschfeld-Smith-Vasilevsky	75
2.	Analysis	77
H.	GROUND 2D.....	83
1.	Hirschfeld-Smith-Fujino	83
2.	Analysis	85
IV.	CONCLUSION.....	87
V.	MANDATORY NOTICES	87
A.	Real Party-In-Interest Under 37 C.F.R. § 42.8(b)(1).....	87
B.	Related Matters Under 37 C.F.R. § 42.8(b)(2)	88
C.	Lead And Back-Up Counsel Under 37 C.F.R. § 42.8(b)(3).....	88
D.	Service Information	88

LIST OF EXHIBITS

MS-1001	U.S. Patent No. 11,656,915 to Pabari (“the ’915 Patent”)
MS-1002	Prosecution History of the ’915 patent (“the Prosecution History”)
MS-1003	Declaration and Curriculum Vitae of Dr. Michael Mitzenmacher
MS-1004	U.S. Patent No. 7,577,722 (“Khandekar”)
MS-1005	U.S. Pat. App. No. 2005/0044301 (“Vasilevsky”)
MS-1006	U.S. Patent No. 6,880,002 (“Hirschfeld”)
MS-1007	U.S. Pat. App. No. 2002/0049803 (“Bandhole”)
MS-1008	U.S. Patent No. 7,457,925 (“Fujino”)
MS-1009	Int. Pat. Pub. No. WO 02/03220 (“Tremain”)
MS-1010	Smith, James E. & Ravi Nair, “Virtual Machines: Versatile Platforms for Systems and Processes,” Morgan Kaufmann, 2005, ScienceDirect (“Smith”)
MS-1011	Declaration of June Munford
MS-1012-1049	RESERVED
MS-1050	Complaint, <i>ToutVirtual, Inc. v. Microsoft Corporation</i> , 2-25-cv-00943, EDTX (filed Sep. 8, 2025)
MS-1051	Docket Control Order, <i>ToutVirtual, Inc. v. Microsoft Corporation</i> , 2-25-cv-00943, EDTX (issued Feb. 19, 2026)
MS-1052	Excerpt of Prosecution History of U.S. Pat. No. 8,799,431

LISTING OF CHALLENGED CLAIMS

Claim 1	
[1.pre]	A method for access to one or more virtualization system resources over a network, the method comprising:
[1.1]	implementing a virtualization system that has a control computer and one or more physical resources and virtual assets connected to the control computer,
[1.2]	wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems, and
[1.3]	the virtual assets are managed by virtualization layer instances running on each of the physical resources;
[1.4]	connecting an end user to the control computer; and
[1.5]	accessing, through the control computer connection, the virtual assets by the end user.
Claim 2	
[2]	The method of claim 1, wherein connecting the end user to the control computer further comprises connecting the end user to the control computer over one of a network, a wide area network or the Internet.
Claim 3	
[3]	The method of claim 2, wherein connecting the end user to the control computer further comprises providing the end user, a desktop, a laptop or a personal digital assistant (PDA) and client resources.
Claim 4	
[4]	The method of claim 3, wherein accessing the virtual assets further comprising accessing an application server by the end user.

Claim 5	
[5]	The method of claim 1, wherein the virtual asset is providing end user access to one of an application server, to a middleware server and to a database.
Claim 6	
[6]	The method of claim 1, wherein the virtual asset is providing a virtual storage resource.
Claim 7	
[7]	The method of claim 6, wherein the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.
Claim 8	
[8]	The method of claim 7, wherein providing the virtual storage resource further comprises mapping a logical storage unit to a physical storage disk.
Claim 9	
[9]	The method of claim 1, wherein providing the virtual assets further comprise providing a logical network resource.
Claim 10	
[10]	The method of claim 9, wherein providing the logical network resource further comprises mapping a logical network unit to a physical network device.
Claim 11	
[11.pre]	A method, comprising:
[11.1]	communicating, by a user, over a network to access one or more virtual assets running in a virtualization system,
[11.2]	wherein the virtualization system includes a control computer and one or more physical resources,

[11.3]	wherein each of the one or more physical resources execute at least two virtual assets that comprise software instantiations of computer systems and
[11.4]	the virtual assets are managed by virtualization layer instances running on each of the one or more physical resources;
[11.5]	requesting, by the user, provisioning of a virtual asset of the one or more virtual assets in the virtualization system; and
[11.6]	wherein the provisioned virtual asset is a virtual storage and the virtual storage comprises one or more logical storage units mapped to one or more physical storage drives.
Claim 12	
[12]	The method of claim 11 further comprising requesting, by a management application programming interface (API) of the virtualization system, the provisioning of the virtual storage.
Claim 13	
[13]	The method of claim 12 further comprising configuring, by the user, the virtual storage.
Claim 14	
[14]	The method of claim 13 further comprising receiving, by the user, an inventory of the virtual storage.

Microsoft Corporation (“**Petitioner**” or “**Microsoft**”) petitions for *Inter Partes* Review (“**IPR**”) of claims 1-14 (“**the Challenged Claims**”) of U.S. Patent No. 11,656,915 (“**the ’915 patent**”). Compelling evidence presented in this Petition demonstrates at least a reasonable likelihood that Microsoft will prevail with respect to at least one of the Challenged Claims.

I. REQUIREMENTS

A. Standing

Microsoft certifies that the ’915 patent is available for IPR and is being filed within one year of service of complaint by Patent Owner. MS-1050. Microsoft is not barred or estopped from requesting this review challenging the Challenged Claims on the below-identified grounds.

B. IPR Challenge

Microsoft requests an IPR of the Challenged Claims on the grounds noted below. Dr. Michael Mitzenmacher provides supporting testimony in his Declaration. MS-1003, ¶¶1-270.

Ground	Claims	Basis for Unpatentability
1A/1B	1, 2	§§102/103: Khandekar
1C	3-5	§103: Khandekar, Bandhole
1D	6-8, 11-14	§103: Khandekar, Vasilevsky

Ground	Claims	Basis for Unpatentability
1E	9, 10	§103: Khandekar, Fujino
2A	1, 2	§103: Hirschfeld, Smith
2B	3-5	§103: Hirschfeld, Smith, Bandhole
2C	6-8, 11-14	§103: Hirschfeld, Smith, Vasilevsky
2D	9, 10	§103: Hirschfeld, Smith, Fujino

The '915 patent claims priority to a series of continuation applications that each claim priority to U.S. Provisional Application No. 60/708,473, filed 8/15/2005 (“**Critical Date**”). MS-1001, Cover. While Microsoft does not concede that the '915 patent is entitled to any of the claims to priority, regardless, each reference asserted in this Petition qualifies as prior art (as shown in the table below).

Reference	Filing Date	Publication Date	Basis
Khandekar	4/5/2002	8/18/2009	§102(e)
Vasilevsky	4/26/2004	2/24/2005	§102(a)
Hirschfeld	3/18/2002	4/12/2005	§102(a)
Bandhole	5/17/2001	4/25/2002	§102(b)

Fujino	6/24/2005	11/25/2008	§102(e)
Smith	NA	7/2005	§102(a)

As evidenced by the librarian declaration (MS-1011), Smith is a non-patent publication that was publicly available prior to the Critical Date at least in the Penn State University Library. MS-1011, ¶¶1-10.

C. Claim Construction

“[C]laim terms need only be construed to the extent necessary to resolve the controversy.” *Wellman, Inc. v. Eastman Chem. Co.*, 642 F.3d 1355, 1361 (Fed. Cir. 2011). In this case, Microsoft submits that the Grounds would render the Challenged Claims obvious under any reasonable construction, including plain and ordinary meaning. For clarity, Microsoft proposes the following constructions, which are consistent with disclosure found in the ’915 patent, and therefore, demonstrate subject matter that falls within the metes and bounds of these terms.

Microsoft does not concede that each challenged claim satisfies all statutory requirements (e.g., §112 requirements), nor does Microsoft waive any arguments concerning claim scope or grounds that can only be raised in a district court. The mappings included in Section III are not inconsistent with these constructions.

1. “control computer”

The '915 patent states that “[t]he Control Center computers are represented in FIG. 1 as physical devices rather than virtual assets...” MS-1001, 5:66-6:35. “Each Control Center computer 110, 112 that is equipped with the virtualization management application described herein will have management of one or more physical assets within a domain or other subnet arrangement associated with the respective Control Center.” *Id.*, 7:5-17. Accounting for context offered by the '915 patent, a POSITA would have understood that the term “control computer” at least encompasses a “computing device that is separate and distinct from the physical host machines being managed and is not itself deployed as a virtual asset.” MS-1003, ¶57.

2. “virtual assets”

The '915 patent specifies that “virtual devices that exist only as software instantiations of equipment objects in a virtual environment will be referred to as virtual assets.” MS-1001, 6:36-41. The '915 patent further explains that “[t]he resource-asset dichotomy will be maintained throughout this discussion.” *Id.* Accounting for context offered by the '915 patent, a POSITA would have understood that the claimed “virtual asset” at least encompasses “software instantiations of equipment objects that exist only within a virtual environment.” MS-1003, ¶¶57-58.

3. “resources”/“physical resources”

The '915 patent specifies that, “[i]n this description, physical devices such as computers, printers, routers, and other ‘boxes’ will be referred to as resources.” MS-1001, 6:36-41. The '915 patent further explains that “[t]he resource-asset dichotomy will be maintained throughout this discussion.” *Id.* Accounting for context offered by the '915 patent, a POSITA would have understood the claimed “resources” and “physical resources” to encompass “physical devices on the network, such as computers, printers, routers, and other tangible hardware.” MS-1003, ¶¶57-58.

4. “network resource”

The '915 patent states that examples of “network resource[s]” include “routers, switches, bridges, infiniband, wireless, radio, optical, fiber channel, link aggregation technologies (such as BitTorrent and the like)[.]” MS-1001, 2:14-30. Accounting for context offered by the '915 patent, a POSITA would have understood that the claimed “network resource” at least encompasses “physical network infrastructure devices, including routers, switches, bridges, infiniband, wireless, radio, optical, and fiber channel equipment.” MS-1003, ¶¶57-58.

5. “software instantiations of computer systems”

The '915 patent states that “[virtual] assets...provide the functional equivalent of desktops (user interfaces), operating systems, applications, servers,

data bases, and the like.” MS-1001, 3:13-24. “In FIG. 3, the virtual assets are illustrated as two virtual computers, or machines, a first Virtual Machine 328 and a second Virtual Machine 330.” *Id.*, 10:1-23. “Each virtual machine can include an operating system (OS) for the virtual machine, computer applications programs executing on the virtual machine OS, and virtual hardware assets for the virtual machine.” *Id.* Accounting for context offered by the ’915 patent, a POSITA would have understood that the claimed “software instantiations of computer systems” at least encompasses “software that can virtually perform functions of physical devices that can connect to computers.” MS-1003, ¶¶57-58.

6. “provisioning”/“provisioned”

The ’915 patent states that the “server virtualization configuration management scheme” of Figure 11 includes “an association or mapping between the actual Physical Resources, such as a collection of computer hardware machines 1104, to a collection of Virtual Machine Assets 106, or applications...” MS-1001, 20:19-39. Accounting for context offered by the ’915 patent, a POSITA would have understood that the claimed “provisioning” and “provisioned” at least encompass “associating a virtual asset to a particular physical computer resource.” MS-1003, ¶¶57-58.

7. “control computer connection”

The ’915 patent states that “[u]sers at network computers 102, 104, 106 can

communicate over a network 108 to gain access to system resources that are available through Control Center computers.” MS-1001, 5:66-6:35. Additionally, “users can be connected directly to a Control Center computer.” *Id.*, 6:64-7:4. “The Control Agent can provide the necessary communications facilities for communications between the host machine and the associated Control Center computer.” *Id.*, 22:10-28. Accounting for context offered by the ’915 patent, a POSITA would have understood that the claimed “control computer connection” at least encompasses “a communication pathway where the control computer acts as an intermediary or gateway in the data path, such that the end user’s access to the virtual assets must be routed directly through the control computer itself.” MS-1003, ¶¶57-58.

8. “logical”

The ’915 Patent states that “FIG. 12 depicts the configuration management 120 of the network computer system 100 (FIG. 1) as comprising an association or mapping between actual Physical Resources comprising a collection of disk drives or storage devices 1204 to a collection of Logical Units 1206, such as network virtual storage drives, in accordance with a set of application critical Rules 1208.” MS-1001, 20:45-21:3. “That is, the storage servers (physical resources comprising disks) are mapped to virtual assets (logical units).” *Id.* Accounting for context offered by the ’915 patent, a POSITA would have understood the claimed “logical”

encompasses “software-defined abstraction that requires mapping to underlying physical hardware.” MS-1003, ¶¶57-58.

9. “physical storage drives”

The ’915 Patent states that “a collection of disk drives or storage drives 1204” may be “assigned to a different network drive assignment, such as drive c:, or d:, or e:...” MS-1001, 20:45-21:3. For example, “a particular network drive should have a capacity of 50 GB...” *Id.* Accounting for context offered by the ’915 patent, a POSITA would have understood the claimed “physical storage drives” encompasses “specific, individual hardware components (e.g., a specific hard disk or a physical router) that are directly addressed and mapped to by the virtualization layer.” MS-1003, ¶¶57-58.

D. Level of Ordinary Skill in the Art

A person of ordinary skill in the art (“POSITA”) relating to the subject matter of the ’915 patent as of the Critical Date would have had (1) at least a bachelor’s degree in computer science, computer engineering, electrical engineering, or a related field, and (2) at least two years of industry experience in virtualization systems. MS-1003, ¶¶20-22. Additional education could substitute for professional experience, and professional experience could substitute for education. *Id.*

II. THE '915 PATENT

A. Specification

The '915 patent describes a “Control Center” that provides “automated management of network-based virtual environments.” MS-1001, 6:42-63. The Control Center “manages multiple resources” (e.g., blade servers and other network-based equipment) that “can be configured to support a variety of virtual assets, including desktop computer environments, data storage, application servers, routers, and the like.” *Id.*, 7:5-35. The Control Center includes modular application components, such as an “Asset Manager” and a “Provisioning Manager,” and includes a “Virtual Mapping engine” that is “programmed to discover the physical resources” and “virtual resources, such as virtual servers, storage, desktops, and the like with which the Control Center has access.” *Id.*, 4:4-15, 7:36-59. The Asset Manager provides “visibility and management of all global virtual assets,” including “a global topology view and an inventory of virtual infrastructure from a central console.” *Id.*, 8:7-20. The Asset Manager supports “inventory reports at local, remote, and global context.” *Id.*; MS-1003, 45-47.

B. Relevant Prosecution

The '915 patent issued on 5/23/2023 from U.S. Patent App. No. 17/074,500 (“the '500 Application”). Examination of the '500 Application included three

Office Actions, all of which presented rejections that included Vasilevsky and U.S. Pat. App. No. 2006/0069761 (“Singh”). MS-1002, 106-24, 173-86, 230-42; MS-1003, 48-56. In addition to Vasilevsky (applied to demonstrate unpatentability of select dependent claim features), the Examiner was aware of Khandekar (which serves the basis for one set of unpatentability challenges in the Petition) during examination of the ’500 Application. *Id.* And yet, the Examiner inexplicably failed to apply Khandekar in any ground of rejection. *Id.* As demonstrated in §III.A.2(a), *infra*, Khandekar discloses the “one or more physical resources” that “execute at least two virtual assets that comprise software instantiations of computer systems” (recited in [1.2]) and “the virtual assets are each managed by virtual layer instances running on each of the physical resources” (recited in [1.3])—which are the full complement of features that Applicant repeatedly urged as distinguishable over the prior art of record (MS-1002, 216-18, 281-82, 292-94), and which consequently led to the Examiner finding the claims to be allowable over the prior art of record. MS-1002, 322.

III. UNPATENTABILITY GROUNDS

A. GROUNDS 1A/1B

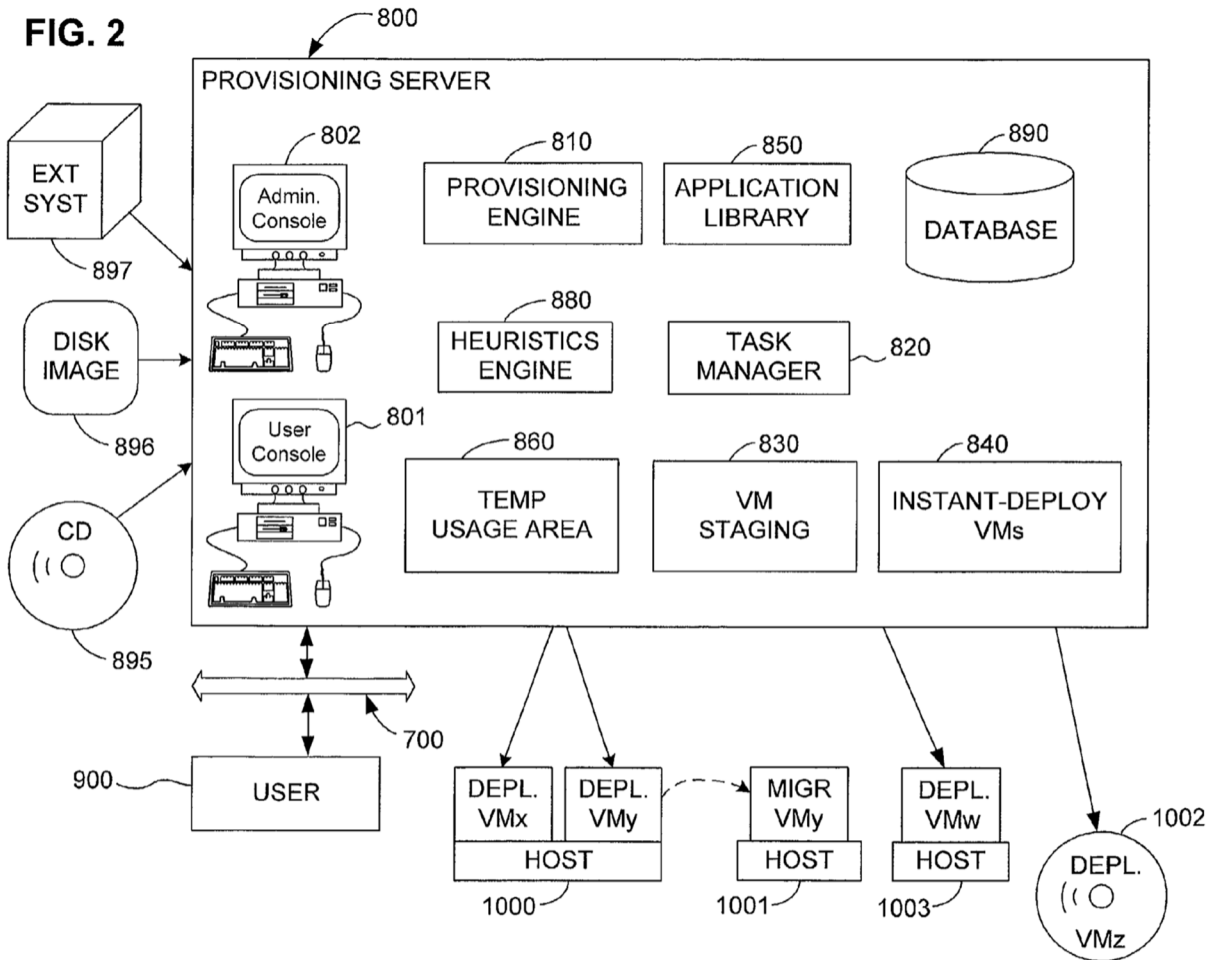
1. Khandekar

Khandekar is directed to a virtual machine provisioning architecture in which a centralized “provisioning server automatically configures a virtual

machine (VM) according to user specifications.” MS-1004, Abst., 9:65-10:4.

Specifically the user “interacts with any conventional web browser” to “custom-provision a VM” or select a “pre-configured, instantly deployable VM.” MS-1004, 13:66-14:3. Khandekar describes that a user submits configuration selections (e.g., OS configuration such as “hostname, IP address, MAC address”), selects desired applications, and then “submits the request to the provisioning server,” which applies heuristics/rules to select an appropriate staged VM and an appropriate host on which to deploy that VM. MS-1004, 9:65-10:42.

Figure 2 shows an example of a system 800 used for provisioning:



MS-1004, FIG. 2.

As shown above, a centralized VM provisioning environment allows one or more users 900 to communicate with the provisioning server 800 via a network 700. MS-1004, 9:46-64. The VM provisioning server 800 includes a front-end user console 801 through which the user can submit the selected options.” *Id.*, 10:15-26. The VM provisioning server may include an administration console 802 “to monitor the operational state” and “maintain the database.” *Id.*, 9:46-64, 10:15-26. The provisioning server receives user input identifying desired

configuration characteristics (e.g., operating system selections, application selections, and “OS configuration, for example, hostname, [and] IP address”) and applies a rules/heuristics component (e.g., heuristics engine 880) to “select the most appropriate VM from a library 830 of staged VMs” and to “select[] which host machine 1000, 1001 is most appropriate to provision and deploy the VM on.” *Id.*, 9:65-10:42.

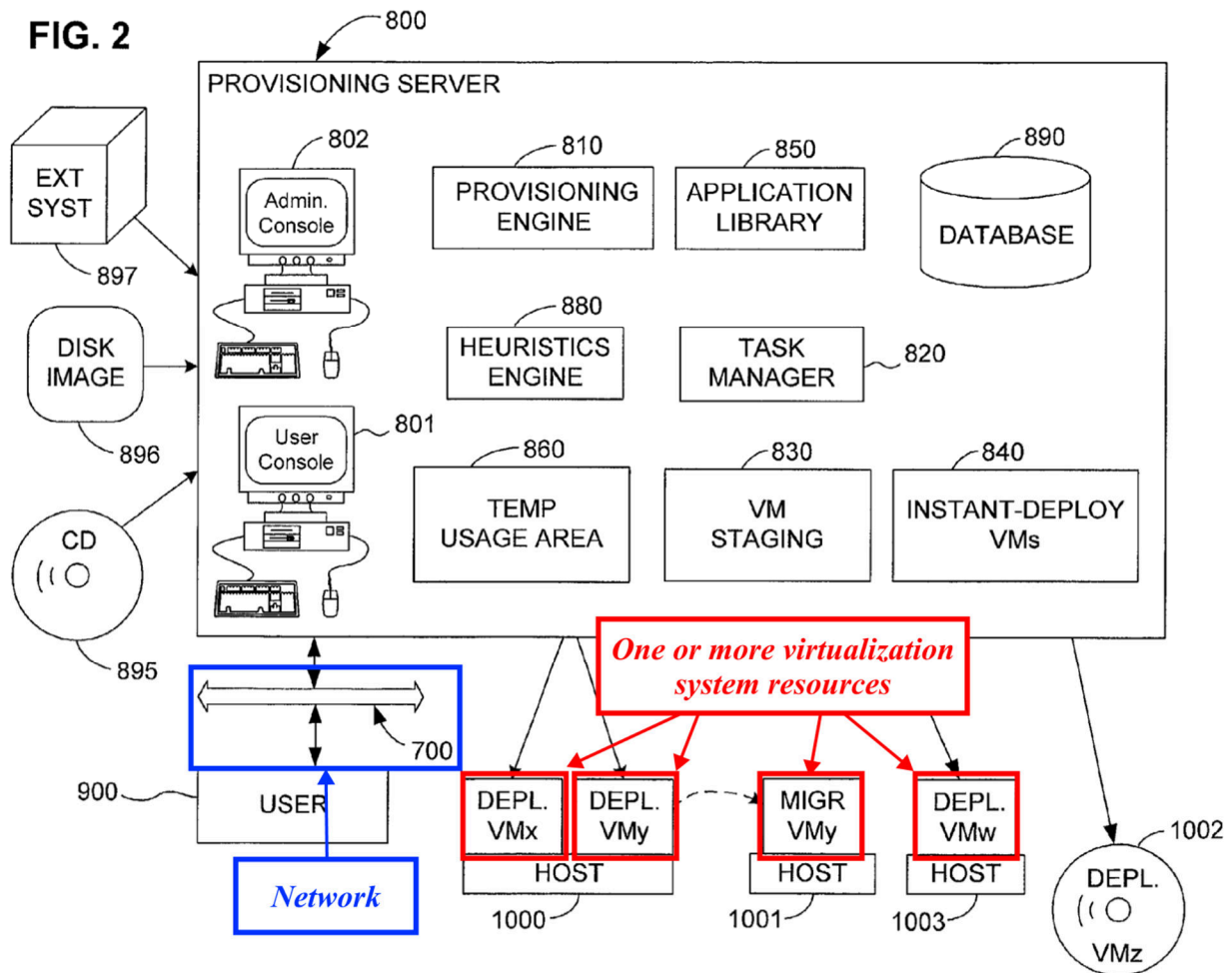
Host systems provide an execution platform for deployed VMs (e.g., VMx and VMy). Moreover, “any number of VMs may be deployed and loaded on a single host,” with VMs capable of being migrated or copied between hosts (e.g., “virtual machine VMy [has] been migrated (or copied) from host 1000 to host 1001”). MS-1004, 10:50-60, 12:57-13:8. Workflow components support provisioning sessions, including a temporary usage area 860 that “holds the user input as well as the various temporary data needed while the provisioning task is in progress.” MS-1004, 10:61-67, 11:1-10. A task manager 820 initiates a “transaction” and “returns a task identifier” that can be used to “find out the progress of the task.” *Id.*

2. Analysis

(a) Claim 1

[1.pre]

To the extent the preamble is limiting, Khandekar provides [1.pre]. In Khandekar's VM provisioning system, a user can request (via a browser-based user console) and access virtual machines deployed on host servers in a virtualized environment. MS-1004, 9:46-64, 13:66-14:3, 14:4-8. The Khandekar System provides a method for access to one or more VM resources (e.g., resources of VMx, VMy) (*one or more virtualization system resources*) over a network 700 (*network*). MS-1004, FIG. 2 (reproduced below). The "*method*" is further described in the claimed features addressed below. MS-1003, ¶131.



MS-1004, FIG. 2¹.

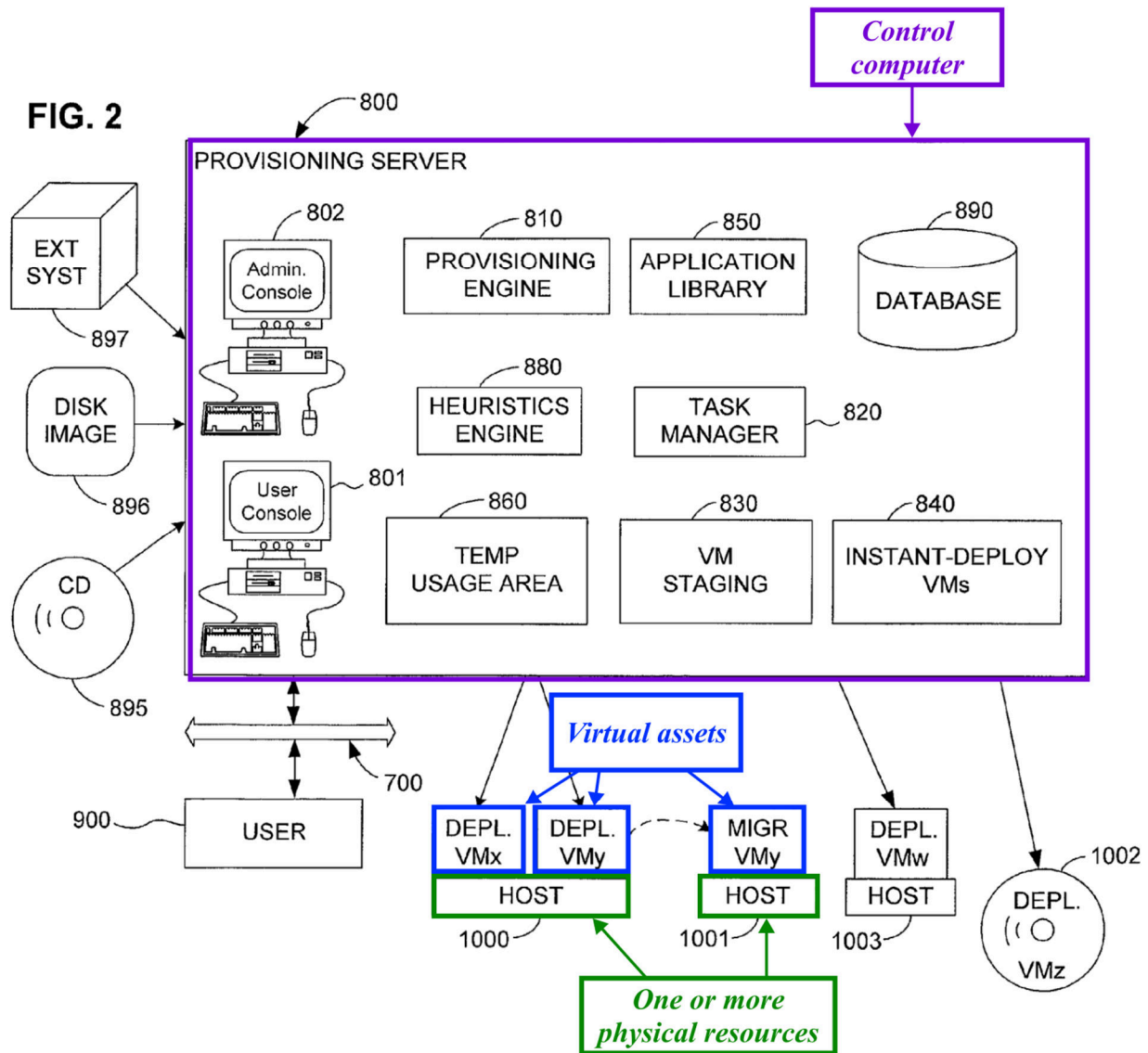
[1.1]

The Khandekar System (*virtualized system*) provides a provisioning server (*control computer*) that communicates over a network with one or more host servers (e.g., 1000, 1001) that provide the underlying hardware compute resources

¹ Annotations to figures throughout this petition are shown in colored markings and text.

(*one or more physical resources*) and execute VMs deployed by the provisioning server (*virtual assets*). MS-1004, 6:1-63, 7:7-27, 9:26-64, 10:27-42, 13:17-35, 17:25-39, 19:33-45, 21:50-52, 21:26-52, 5:35 (describing a host server as a “physical platform”); MS-1003, ¶132.

As noted for [1.pre], Khandekar’s VM provisioning server 800 manages a plurality of “Host” computers, which are the hardware machines on which virtual machines are deployed. MS-1004, 9:26-32, 10:27-42. A virtual machine monitor (VMM) is installed on each of a plurality of hardware hosts and “virtualizes all, or at least some of, the resources of the machine.” *Id.*, 7:7-27, 13:17-35. The VMM is “usually a thin piece of software that runs directly on top of the host,” thereby executing on the physical host hardware. *Id.* In this configuration, the host servers are examples of the recited *physical resources* at least because they are physical platforms that provide the underlying compute infrastructure. MS-1004, 5:35; MS-1003, ¶133. Further, the deployed VMs represent the *virtual assets* because they are instantiated and executed on those hosts under control of the provisioning server. MS-1003, ¶133; MS-1001, 16:31-62.



MS-1004, FIG. 2.

Further, in Khandekar, the provisioning server 800 communicates with the host machines over a network 700 and deploys VMs to selected hosts in response to user requests. MS-1004, 6:1-12, 9:46-64, 10:27-42. For example, when a VM is provisioned, the provisioning engine “deploys a VM on a host” and calls an API on the host to register the new VM, thereby associating the VM instance with the

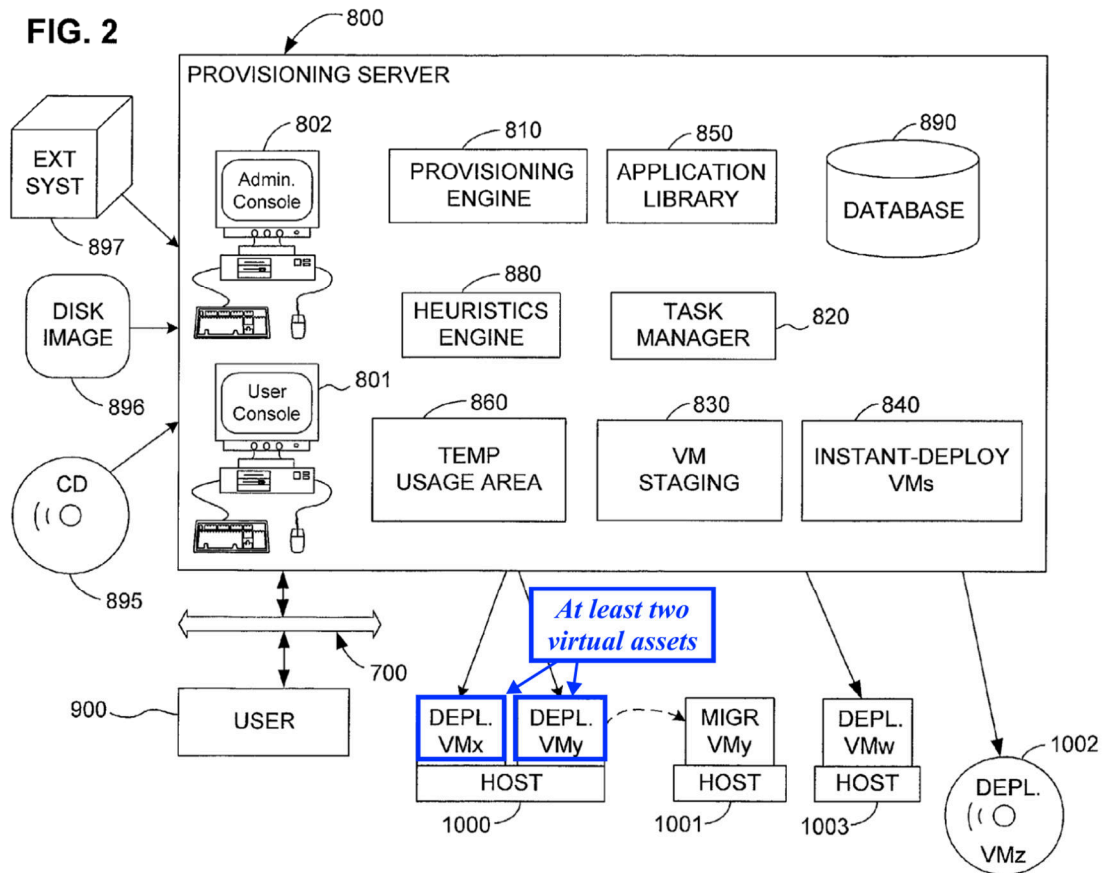
selected physical host. *Id.*, 19:33-45, 21:50-52, 21:26-52. Because provisioning server 800 initiates and manages deployment, configuration, and lifecycle operations of the VMs on the host machines, the VMs are connected to and controlled by the provisioning server 800. MS-1003, ¶¶134-135.

[1.2]

Each of the Khandekar System's host servers 1000, 1001 (*physical resources*) execute at least two VMs (*at least two virtual assets*) that include software that can virtually perform functions of physical computer systems (*software instantiations of computer systems*). MS-1004, 6:24-61, 7:44-47; MS-1003, ¶¶136-139.

As noted in [1.1], Khandekar describes a “virtual machine (VM)” as a “software abstraction” of a “physical computer system.” MS-1004, 6:24-42. In Khandekar, VMs include software constructs that emulate connected physical hardware, such as “optional virtual peripheral devices 340 (VDEVICES)” that are “implemented in software to emulate the corresponding components of an actual computer.” *Id.* As another example, the system includes “device emulators 540 that emulate the characteristics of respective physical devices.” *Id.*, 7:48-55. Thus, at least by providing software that emulates the characteristics and functions of computer systems, e.g., physical devices that connect to computers, the VMs provided by the Khandekar System include software that are examples of *software*

instantiations of computer systems. MS-1003, ¶137.



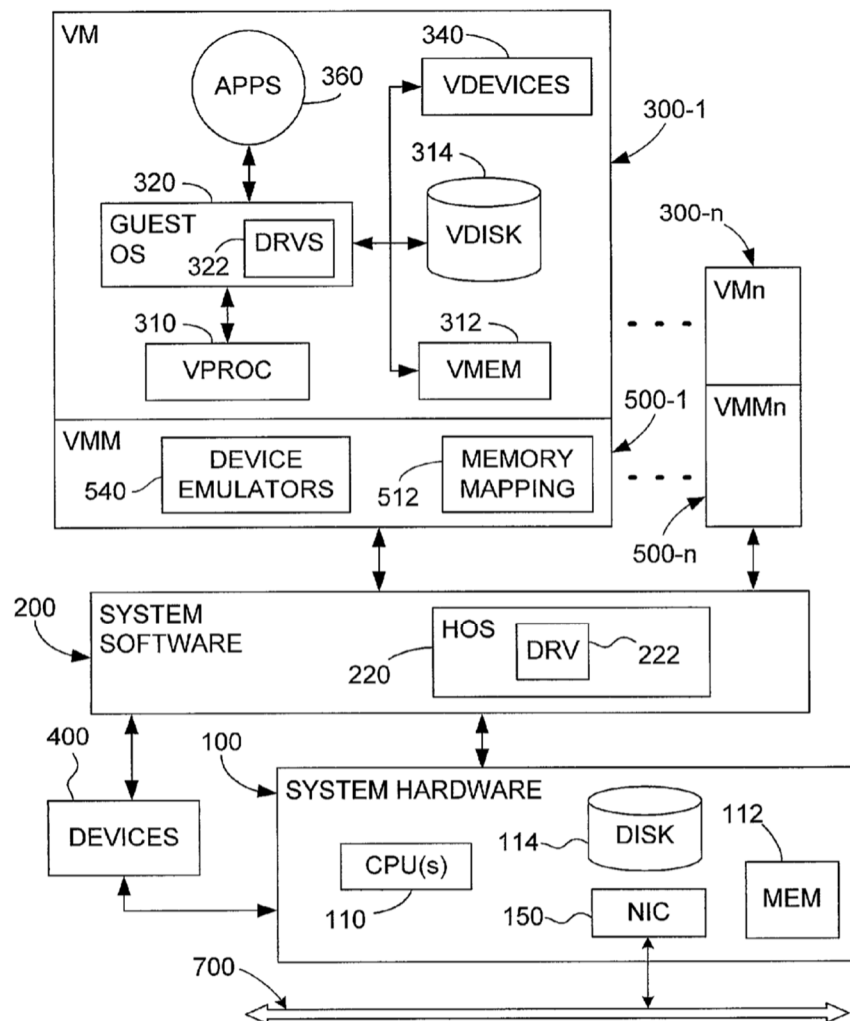
MS-1004, FIG. 2.

Notwithstanding the above, it would have been obvious to a POSITA that in some implementations of the Khandekar System each host server executes at least two VMs. MS-1003, ¶138. For example, if host server 1000 is the “one or more physical resources,” then its execution of VMx and VMy (“*at least two virtual assets that comprise software instantiations of computer systems*”) satisfies [1.2]. MS-1003, ¶138.

In another example, when identifying a prior art configuration in Figure 1,

Khandekar discloses “a **plurality** of virtual machines (VMs)” 300-1 to 300-n that run on the system software 200, thereby revealing that configurations in which multiple VMs are run on a single host server (hardware platform 100) were known as of the Critical Date. MS-1004, 5:52-57; MS-1003, ¶139.

FIG. 1
(Prior Art)



MS-1004, FIG. 1.

[1.3]

The Khandekar System's host servers (e.g., 1000, 1001) (*physical resources*) execute respective VMMs (*virtualization layer instance*) that manage the virtual machines or logical servers (*virtual assets*) executing on that host server. MS-1004, 6:63-7:6, 7:7-27, 13:17-35; MS-1003, ¶¶140-141.

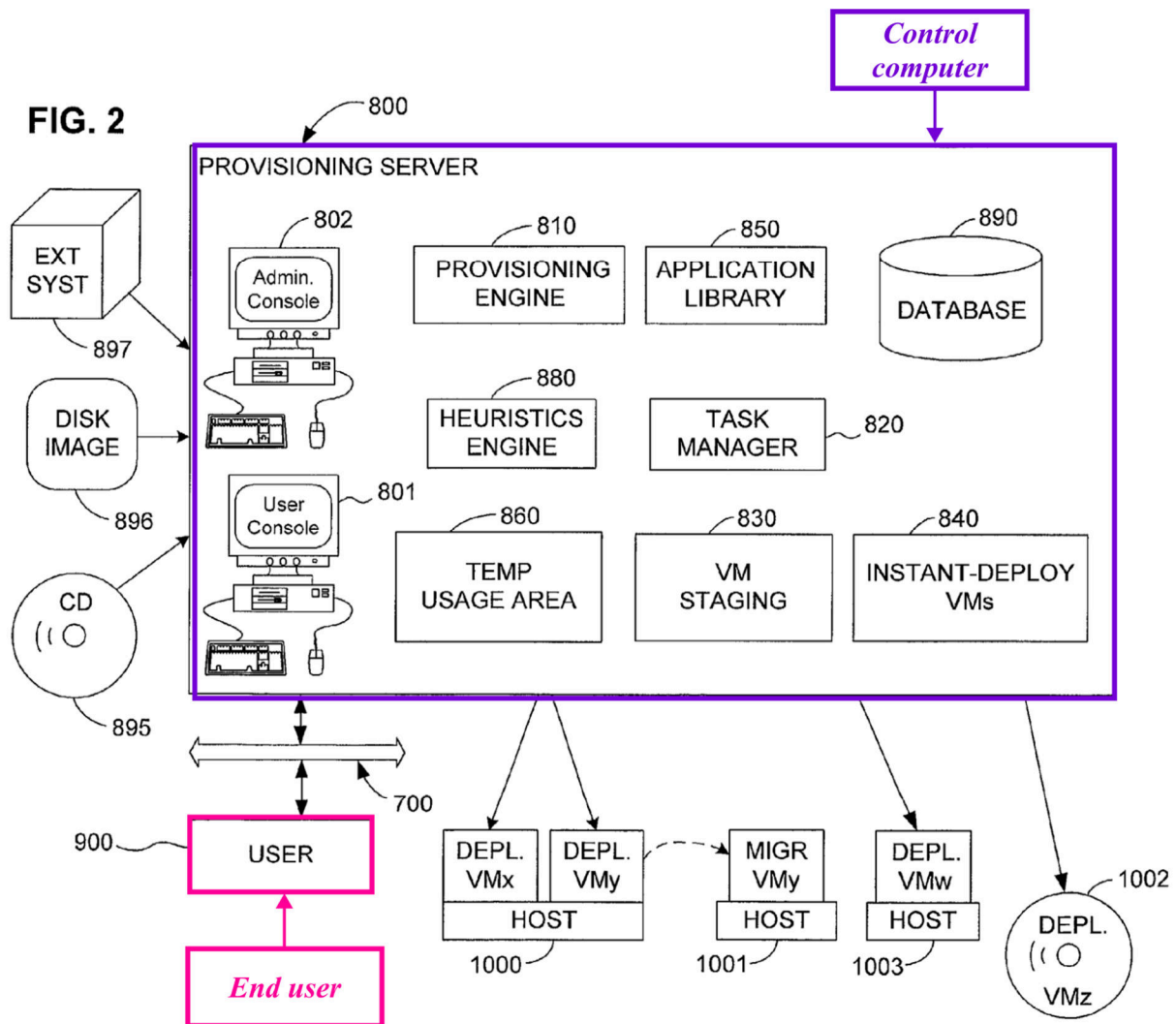
As noted in [1.1] and [1.2], from Khandekar, a “virtual machine monitor (VMM)” is “pre-installed on each host” within a “plurality of hardware hosts.” MS-1004, 6:63-7:6, 13:17-35. The VMM “virtualizes all, or at least some of, the resources of the machine,” and is “usually a thin piece of software that runs directly on top of the host.” *Id.*, 7:7-27. In this architecture, the VMM running on a given host represents a software layer that manages the virtual machines executing on that host by e.g., enabling and maintaining the operating environment on which the VM is executed. MS-1003, ¶141.

[1.4]

The Khandekar System connects a user (e.g., *end user*) via a web browser to the provisioning server (*control computer*) through a user console (e.g., user console 801), thereby providing the user a networked connection to the provisioning server for submitting requests and interacting with it. MS-1004, 1:44-63, 9:46-64, 13:66-14:3; MS-1003, ¶¶142-143.

As noted in [1.pre], Khandekar describes that an end user interacts with the

provisioning system through a user console, which is “preferably a web server, with which the user interacts with any conventional web browser.” MS-1004, 13:66-14:3. The user “logs in to the provisioning server,” and, through the user console, selects a pre-configured virtual machine or submits configuration selections for a custom-provisioned VM. *Id.* This browser-based interaction establishes a network connection between the user and the provisioning server, thereby connecting the user to the provisioning server. *Id.*, 1:44-63, 9:46-64; MS-1003, ¶143.



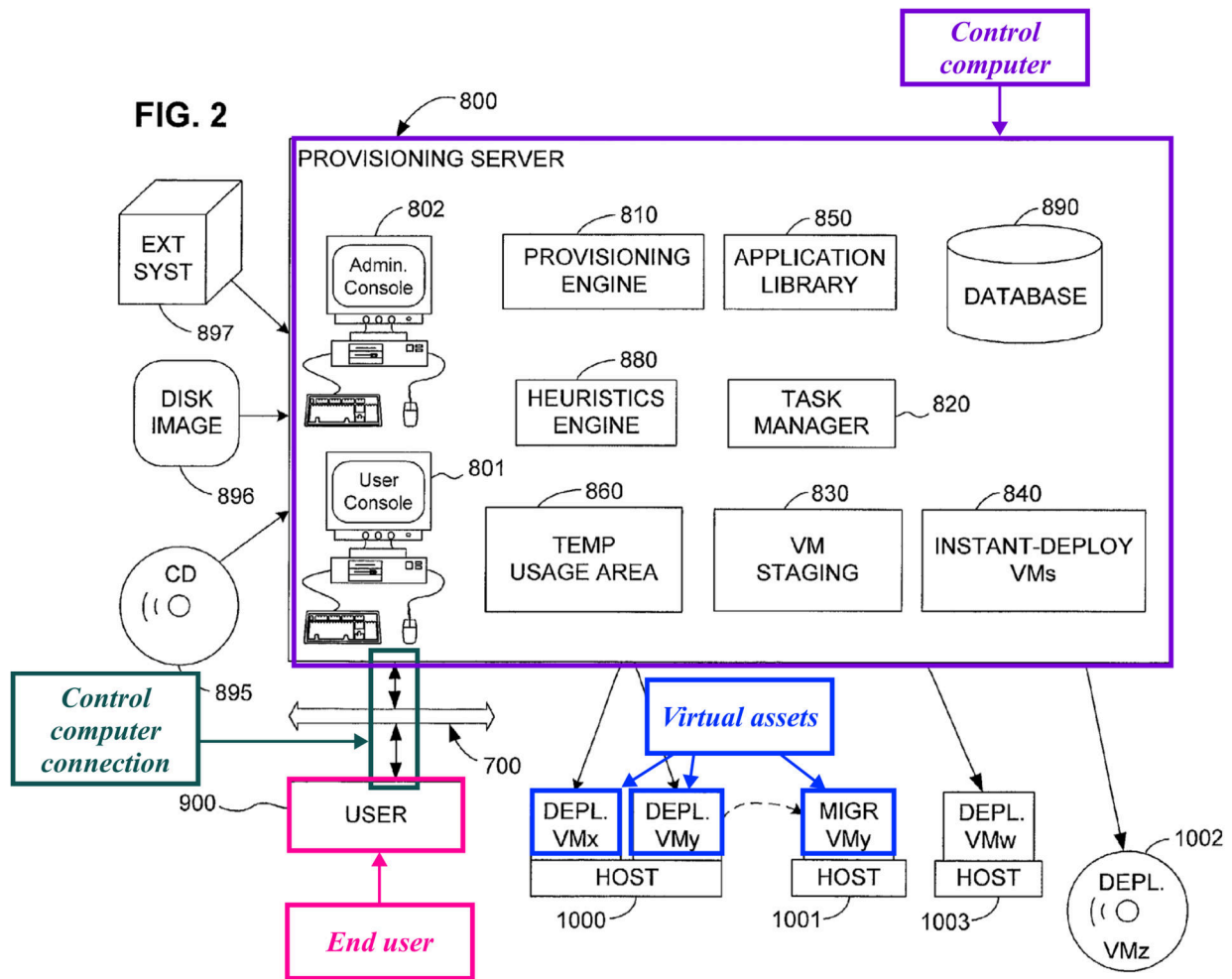
MS-1004, FIG. 2.

[1.5]

The Khandekar System enables the end user to access the VMs (*virtual assets*) deployed on and executed by the host servers (*physical resources*) through the user's connection to the provisioning server (*control computer*) via the user console. MS-1004, 9:65-10:14, 13:66-14:18, 15:29-39, 20:64-21:8, 21:9-15; MS-1003, ¶¶144-147.

As noted in [1.4], in Khandekar, the user connects to the provisioning server through the user console, which is “preferably a web server...” MS-1004, 13:66-14:3. The Khandekar System thereby enables access to VMs through a connection provided by the provisioning server (e.g., 800) (e.g., by logging into the server directly). This connection provides, for example, a communication pathway where the provisioning server acts as an intermediary or gateway in the data path, such that the user’s access to the VMs is routed directly through the provisioning server. MS-1003, ¶145. For example, through this connection, the user may access VM-related resources by selecting a pre-configured VM (e.g., an “instantly deployable” VM) or submitting configuration selections (e.g., OS configurations such as hostname, IP address, and virtual hardware parameters) for a custom-provisioned VM. *Id.*; MS-1004, 9:65-10:14, 14:4-18, 11:20-58.

After submission, the system returns a task identifier and the user can use the console/browser connection to query the status of the provisioning task and, in some cases, abort the task before completion. *Id.*, 15:29-39, 20:64-21:8, 21:9-15. Once the VM is created, the user is informed (e.g., “via the browser from which he submitted the request”). *Id.*, 15:29-39.



MS-1004, FIG. 2.

(b) Claim 2

The Khandekar System connects a user 900 (*end user*) to the provisioning server 800 (*control computer*) through a network 700, including implementations where the network is the *Internet*. MS-1004, 9:46-64. In Khandekar, “[o]ne or more users 900 communicate with the provisioning server 800 via...a network 700, such as the Internet, or over any other known channel.” MS-1004, 9:46-64; MS-1003, ¶148.

B. GROUND 1C

1. Khandekar-Bandhole

(a) Bandhole

Bandhole describes a remotely administered “dynamic computing environment” (DCE) framework that is focused on “configuring, provisioning, accessing and managing” such environments through user interfaces. MS-1007, [0032], [0033]. These interfaces “can be used interactively through a graphical user interface such as a web page or non-interactively through a program script,” and that the framework that permits “‘virtual’ hosting” in which “neither the requisite devices nor the network need to be physically accessible to users.” *Id.* This enables users to “remotely configure, provision, access, and manage DCEs” and to initiate or terminate hosting “from any geographic location.” *Id.*

Bandhole recognizes that computing devices “can be made in many dimensions and styles (e.g., laptop, palmtop, server, workstation, mainframe),” and that “any hardware platform” suitable for the processing may be used. MS-1007, [0035]. A client connects to a web server via the Internet that uses “Java Servlets and JSP” to “receive requests from any web client, such as a browser,” “forward such requests to the application server 311,” and “return the results” back to the client. *Id.*, [0048]. The application server operates to present to the user “methods to create a DCE,” “methods to allocate physical resources required for a DCE,”

and methods to “manage and to interact with a DCE as well as its components.”

Id., [0051]; MS-1003, ¶¶74-78.

(b) Combination

The Khandekar-Bandhole System would have predictably combined Khandekar’s VM provisioning environment with the examples of known user device types from which a user may access a browser-based, remote interface. MS-1003, ¶¶95-97. Khandekar provides a provisioning environment in which one or more users communicate with a provisioning server through a user console that is “preferably a web server” accessed using “any conventional web browser,” while leaving the specific client device type used to run that browser as an implementation detail. MS-1004, 9:46-64, 13:66-14:3. Bandhole confirms that such client-side access was well-known by the Critical Date and that computing devices used in this manner “can be made in many dimensions and styles,” including “workstation,” “laptop,” and “palmtop” form factors. MS-1007, [0035]; MS-1003, ¶95.

A POSITA implementing Khandekar’s browser-based provisioning workflow would have been motivated to enable use of the user console from these known client device types because Khandekar’s approach presupposes remote browser access over a network and does not limit the client platform. MS-1004, 9:46-64, 13:66-14:3; MS-1003, ¶¶96-97. Bandhole’s teachings provide relevant

and compatible examples confirming that remote, web-based interaction with server-side functions may be performed from different user device form factors (e.g., workstation, laptop, palmtop). MS-1007, [0029], [0035], [0048]; MS-1003, ¶¶76, 96-97.

A POSITA would have further recognized that implementing Khandekar's browser-based console access on these known device types is complementary to (and supportive of) Khandekar's self-service provisioning paradigm, without requiring changes to Khandekar's underlying provisioning sequence on the server. MS-1003, ¶97. In this regard, a POSITA would have reasonably expected success because the combination involves applying known client device form factors (e.g., workstation, laptop, palmtop) to a known browser-based access model already contemplated by Khandekar. *Id.* In at least this way, the combination of Khandekar and Bandhole would have merely involved combining prior art elements according to known methods to yield predictable results. *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 415-421, 82 USPQ2d 1385, 1395-97 (2007).

2. Analysis

(a) Claim 3

The Khandekar-Bandhole System would have permitted a user to access VMs through a connection established using a user interface of the provisioning server (e.g., an interface of the provisioning server) (*connecting the end user to*

the control computer). MS-1004, 9:65-10:14, 13:66-14:18, 15:29-39, 20:64-21:8, 21:9-15. A POSITA would have found it obvious that the user device for accessing the VMs may be one of several well-known device types, including a “workstation” (*desktop, client*), “laptop” (*laptop*), “palmtop” (*personal digital assistant (PDA)*)). MS-1007, [0035]; MS-1003, ¶¶149-150.

As noted in [1.5], Khandekar would have enabled access to VMs through a connection provided by the provisioning server 800. MS-1004, 13:66-14:3. A POSITA would have understood and/or found it obvious that the device used to access the VMs may be one of several types of known user device types. *Id.*; see MS-1003, ¶150. For example, Khandekar explains that the “user console is preferably a web server,” with which the user interacts using “any conventional web browser.” MS-1004, 13:66-14:3. It would have been understood by a POSITA that any conventional client computing device capable of running such a web browser and communicating over the network may be used to access the VMs. MS-1004, 13:66-14:3; MS-1003, ¶150.

Bandhole similarly demonstrates that it was well-known by the Critical Date to use various types of user device types to access virtual systems similar the VMs/logical servers described in Khandekar. MS-1007, [0004], [0047], [0048]. In Bandhole, such user device types are described to be made in many “dimensions and styles,” including “workstation,” “laptop,” and “palmtop” form factors. *Id.*,

[0035]. A POSITA would have found it obvious that in the Khandekar-Bandhole System access to VMs would have been provided using any suitable user device, such as a “workstation” (*desktop, client*), “laptop” (*laptop*), “palmtop” (*personal digital assistant (PDA)*). MS-1003, ¶150.

(b) Claim 4

As noted in [2], in the Khandekar-Bandhole System, a user would have accessed VMs through a connection established using a user interface of the provisioning server (e.g., an interface provided by the provisioning server) (*accessing the virtual assets*). MS-1004, 9:46-64, 14:4-18, 23:11-20; MS-1003, ¶¶151-53.

Based on the teachings of Khandekar and Bandhole, a POSITA would have found it obvious that the accessed VM involves, in some implementations, accessing an *application server*. MS-1003, ¶¶151-152. For example, Khandekar suggests a configuration in which an accessed VM represents an application server through disclosure of a web application that uses “an application server on another [computer].” MS-1004, 23:11-20; MS-1003, ¶¶152-153. Thus, in some implementations, the Khandekar-Bandhole System would have provided access to VMs that provide application server functionality such that the access involves *accessing an application server by the end user*. MS-1003, ¶¶152-153.

(c) Claim 5

As noted in [4], the Khandekar-Bandhole System would have enabled a user to access a VM (*virtual asset*) through a connection to a provisioning server, and it would have been obvious that, in some implementations, the accessed VM provides *application server* functionality at least because it is configured with application software. MS-1004, 23:11-20; MS-1003, ¶¶154-155. Further, Khandekar describes that a VM may run an application representing an “database server” thereby enabling access to a *database* when the VM runs a database server application. MS-1004, 11:50-58, 23:11-20. Moreover, because of the phrase “one of,” satisfaction of claim 5 does not mandate providing end user access to each of the enumerated elements recited in the claim.

C. GROUND 1D

1. Khandekar-Vasilevsky

(a) Vasilevsky

Vasilevsky describes a virtualization-management approach directed to dynamic (i.e., runtime) provisioning and reconfiguration of compute resources in a virtual computing system. MS-1005, [0150], [0179]. In particular, Vasilevsky replaces static resource allocation with a “capability for dynamically reconfiguring resources” such that “resources may be allocated (or deallocated) transparently to the applications.” *Id.*, [0179]. To enable that runtime control, Vasilevsky describes a management architecture that allows configuration changes among

“multiple entities (administrators, programs, etc.)” and provides “capability for dynamically reconfiguring resources” such that “resources may be allocated (or deallocated) transparently to the applications.” MS-1005, [0084], [0179]. The management agent maintains a “mapping database that defines mapping of virtual resources to physical hardware,” including mapping a “virtual processor” to a “physical processor,” and, in some implementations, virtual processors and partitions that “can span” physical nodes. *Id.*, [0091], [0094], [0150], [0179].

Vasilevsky further explains that such capabilities may be implemented via a management server/agent arrangement in which the management server writes configuration information (e.g., indicating how each node should be configured) to a data store that management agents read to configure distributed server/virtual server execution. MS-1005, [0085], [0088], [0155]. For resource assignment, a manager may be adapted to perform an association between “unallocated actual resources” and virtual servers. *Id.*, Cl. 84. The system maintains a “mapping database that defines mapping of virtual resources to physical hardware” and defines virtual resources backed by assigned actual resources (e.g., mapping a “virtual network interface” to an “actual network interface”). *Id.*, [0150]; MS-1003, ¶¶79-84.

(b) Combination

A POSITA would have predictably combined Khandekar’s VM provisioning

environment with Vasilevsky's techniques for mapping virtual storage resources to physical hard disks. MS-1003, ¶¶98-103. Khandekar provides a provisioning environment in which a provisioning server receives user requests through a console and uses dynamically maintained host state information to select an appropriate host. MS-1004, 12:57-13:8, 20:54-60. Vasilevsky describes a manager (e.g., manager 212) that is "adapted to perform a mapping" between a "virtual resource" (e.g., "virtual storage entity") and "one or more actual resources" (e.g., "actual storage device"). MS-1005, [0040].

In the combination, Khandekar's provisioning server environment would have been incorporated Vasilevsky's management capabilities (e.g., manager 212) so that VM instances may be implemented as, or be configured to provide, virtual storage capabilities. MS-1004, 12:57-13:8, 20:54-60; MS-1005, [0070], [0072].

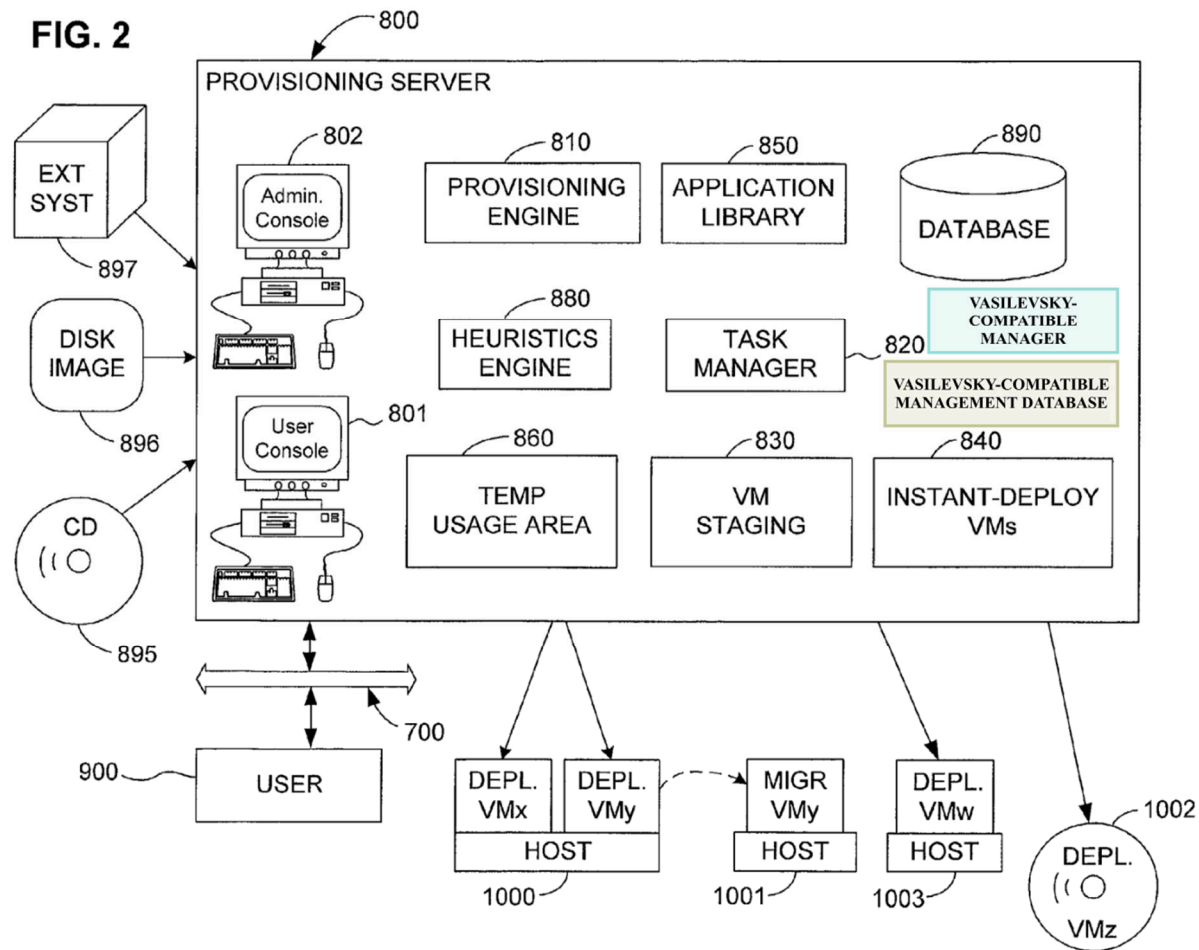
A POSITA implementing Khandekar's provisioning workflows in larger-scale or more dynamic virtualized environments would have been motivated to seek techniques focused on virtual storage capabilities. MS-1003, ¶¶98-103. This is because Khandekar itself contemplates at least a "virtual disk 314 (VDISK)" (as well as other shared storage) as part of its VM but leaves the mapping to actual physical drives as an implementation detail. MS-1004, 6:24-42. A POSITA would have sought well-known storage mapping techniques in implementing the virtual storage functionality described in Khandekar, such as providing "a shared storage

area” for “host servers where the VMs are deployed.” *Id.*, 16:64-17:13, 19:20-25; MS-1003, ¶¶98-103. This would have led the POSITA to Vasilevsky, which identifies “Network Attached Storage (NAS) functions” as being conventionally provided “as a software application executing on a virtual server.” MS-1005, [0011]; MS-1003, ¶¶98-103. Vasilevsky observes that “network attach storage (NAS) or a storage area network (SAN) as are well-known in the art.” *Id.*, [0166].

In one example implementation, Khandekar’s provisioning server 800 is configured to include (or operate in conjunction with) a Vasilevsky-compatible manager (e.g., manager 212) and associated management database (e.g., management database 213) to support virtual storage capabilities. MS-1004, 9:26-32, 13:44-56; MS-1005, [0040], [0150], [0166].

In the Khandekar-Vasilevsky System, a user submits a provisioning request through Khandekar’s user console 801 (e.g., selecting an instantly deployable VM or specifying a VM configuration that includes disk parameters), and the provisioning server selects a host and requests the host (via an API) to reserve disk space required for the VM. MS-1004, 13:66-14:3, 14:31-64, 20:54-60. As part of (or following) these disk reservation and VM deployment operations, the Vasilevsky-compatible manager (within the provisioning server environment) performs and/or maintains a “mapping” between the VM’s storage-related virtual resource (e.g., “virtual storage entity”) and the corresponding “actual resource”

(e.g., “actual storage device”), consistent with Vasilevsky’s disclosure that the manager is “adapted to perform a mapping” between them. MS-1005, [0040]. In this way, alongside Khandekar’s task-based provisioning flow, the Khandekar-Vasilevsky System maintains mapping information that records how the VM’s virtualized storage construct corresponds to underlying physical storage, thereby supporting virtual storage capabilities. MS-1004, 20:54-21:24; MS-1005, [0040], [0150].



MS-1004, FIG. 2 (modified to incorporate Vasilevsky’s teachings).

As explained above, incorporating Vasilevsky's teachings for representing and controlling how virtualized constructs correspond to underlying physical resources (including during execution) would have represented an implementation detail for Khandekar's teachings regarding "virtual disk 314 (VDISK)" as part of its VMs. MS-1004, 6:24-42. A POSITA would have therefore reasonably expected success in combining the teachings of Khandekar and Vasilevsky since the combination amounts to using known virtualization-management techniques to enhance a known VM provisioning workflow that already operates through well-defined management interfaces. MS-1003, ¶¶98-103.

2. Analysis

(a) Claim 6

The Khandekar-Vasilevsky System would have provided a VM (*virtual asset*) configured to include, utilize, or present a virtualized storage construct (e.g., "virtual storage entity"), thereby providing an example of a *virtual storage resource*. MS-1004, 6:24-42; MS-1005, [0150], [0156], [0172].

As discussed in Grounds 1A/1B, Khandekar's VM includes a "virtual disk" (e.g., 314) that stores the VM's operating system and other installed software. MS-1004, 6:24-42. A POSITA would have recognized and/or found obvious that the VM may be implemented to utilize advanced virtual storage configurations, such as the virtualized resources described in Vasilevsky. MS-1003, ¶158. In

Vasilevsky, a virtual resource includes a “virtual storage entity” that is distinct from an “actual storage device” (e.g., Vasilevsky's “disks 914”). MS-1005, [0166], Cl. 80. Thus, in the Khandekar-Vasilevsky System, the VM (*virtual asset*) provides a “virtual storage entity” as a *virtualized storage resource* presented as part of, or to support, the virtualized server instance. MS-1003, ¶¶157-158.

(b) Claim 7

As noted in [6], the Khandekar-Vasilevsky System would have provided a VM (*virtual asset*) that includes a virtualized storage construct, such as a “virtual storage entity” corresponding to physical storage. Consistent with Vasilevsky, in the combination, the virtual storage construct includes a mapping between (i) a storage abstraction presented logically to a virtualized system (*logical storage unit*) and (ii) a hardware component (*physical storage disk*). MS-1005, Cls. 78, 84, 85.

Vasilevsky describes maintaining a “mapping” between a “virtual resource” and “actual resources” to enable allocation and management of resources for virtualized computing instances. MS-1005, Cl. 78. The actual resources include hardware components, such as “one or more storage devices (e.g., disks 914).” *Id.*, [0166]. The system includes a “mapping database that defines mapping of virtual resources to physical hardware.” *Id.*, [0150]. A POSITA would have understood that this mapping database permits the virtualization layer to map the logical

storage abstractions directly to these specific, individual hardware disks. MS-1003, ¶¶159-160. The '915 patent specification describes that a logical storage construct (e.g., a logical storage unit) is mapped onto underlying storage hardware (e.g., disk drives or storage devices). MS-1001, 20:45-21:3.

(c) Claim 8

See [7]; MS-1003, ¶161.

(d) Claim 11

[11.pre]

See [1.pre]; MS-1003, ¶162.

[11.1]

See [1.pre]. The Khandekar-Vasilevsky System would have enabled a user (e.g., 900) to **communicat[e]** over a network (e.g., network 700) (**network**) with the provisioning server and portal interface to access one or more VMs (e.g., VMx, VMy) (**virtual assets**) operating on host servers in the virtualization environment (**virtualization system**). MS-1004, 9:46-64, 13:66-14:3, 14:4-18; MS-1003, ¶163.

[11.2]

See [1.1]; MS-1003, ¶¶164-166.

[11.3]

See [1.2]; MS-1003, ¶¶167-168.

[11.4]

See [1.3]; MS-1003, ¶¶169-170.

[11.5]

The Khandekar-Vasilevsky System would have received, from a user communicating via a console (e.g., 801) (*requesting, by the user*), a request to provision a selected VM (e.g., VMx) (*provisioning of a virtual asset*) within the system (*virtualization system*). MS-1004, 9:46-64, 13:66-14:3, 19:63-20:2, 20:64-21:8; MS-1003, ¶¶171-172.

As noted in [1.4] and [1.5], in Khandekar, an end user interacts with provisioning server 800 through user console 801 using “any conventional web browser.” MS-1004, 9:46-64, 13:66-14:3. In this workflow, “[w]hen the user submits a request to provision a new VM, the provisioning engine requests that a new task be created,” and the task manager returns a task identifier for the requested provisioning operation. *Id.*, 20:64-21:8. The user may request provisioning by, for example, selecting a pre-configured VM or providing configuration inputs (e.g., “computer hostname, IP address, etc.”) as part of the provisioning request submitted through the console connection. *Id.*, 13:66-14:3, 19:63-20:2. Thus, in the Khandekar-Vasilevsky System, the user may request provisioning of a virtual asset (a new VM) in the virtualization system. MS-1003, ¶¶171-172.

[11.6]

In the Khandekar-Vasilevsky System, a selected VM provisioned based on a user provisioning request (*provisioned virtual asset*) may be configured to function as a storage server for the network, and thus, acting as an example of a *virtual storage resource*. MS-1005, [0011], [0150], [0166]; MS-1003, ¶¶173-175.

A POSITA would have understood that virtual machines are routinely provisioned to provide storage capabilities because Vasilevsky, for example, describes how specialized storage hardware is replaced by a “software-only application that is executed on a virtual server,” e.g., by providing “Network Attached Storage (NAS) functions.” MS-1005, [0011]; MS-1003, ¶¶173-175. Thus, in some implementations of the Khandekar-Vasilevsky System, the provisioned VM functions as a *virtual storage*.

Moreover, to provide storage capabilities, the provisioned VM may further utilize underlying virtualized storage constructs. MS-1003, ¶¶173-175. Consistent with Vasilevsky, the system manages these constructs by maintaining a mapping between (i) a storage abstraction presented logically to the virtualized system (e.g., Vasilevsky’s “virtual storage entity”) (*one or more logical storage units*) and (ii) underlying hardware storage devices (e.g., Vasilevsky’s “disks 914”) (*one or more physical storage drives*). MS-1005, [0150], [0166]. In such configurations, the provisioned VM operates as an example of *virtual storage* in that it maps logical storage to physical storage drives. MS-1003, ¶¶173-175.

(e) Claim 12

The Khandekar-Vasilevsky System would have enabled a user to request provisioning operations through a programmatic interface of the provisioning server (e.g., 800) (*application programming interface (API) of the virtualization system*). This includes provisioning operations directed to storage components of a provisioned VM (*provisioning of the virtual storage*). MS-1004, 9:46-64, 22:53-67; MS-1003, ¶¶176-178.

Khandekar describes that provisioning may be requested by a user, and alternatively, by “a software application or tool.” MS-1004, 9:46-64. In such implementations, “[s]oftware applications acting as a user interact with the provisioning server using a well-defined programmatic interface (API)...” *Id.* Thus, in implementations of the Khandekar-Vasilevsky, provisioning requests are issued via an API that forms part of the provisioning server’s control-plane interface for provisioning operations. *Id.*, 9:46-64, 22:53-67. Further, as noted in [11.6], the Khandekar-Vasilevsky System would have enabled provisioning of a selected VM that includes a virtualized storage construct, such as a “virtual storage entity” corresponding to physical storage (*virtual storage*). MS-1003, ¶¶176-178.

(f) Claim 13

See [6]; MS-1003, ¶179.

(g) Claim 14

The Khandekar-Vasilevsky System would have enabled a user (via a management-plane interaction used for requesting provisioning) to *receiv[e]* a listing of available virtual storage resources (*inventory of the virtual storage*) presented for selection/assignment in connection with provisioning operations. MS-1004, 10:15-26, 11:1-10, 13:66-14:3, 15:29-39; MS-1005, [0084], [0150].

As noted in [13], Khandekar provides the user-facing management pathway (e.g., a user console accessed via a web browser) through which a user submits requests and receives system-provided information (e.g., selectable options and task-related feedback). MS-1004, 10:15-26, 11:1-10, 13:66-14:3, 15:29-39. Vasilevsky, in turn, describes that a manager may “perform an association” between “unallocated resources” and virtual servers, allowing configuration access among “multiple entities (administrators, programs, etc.).” MS-1005, [0084]. In the context of Vasilevsky’s virtualization management, this association of unallocated resources would include storage-related virtual resources (e.g., virtual storage entities) corresponding to underlying physical storage devices under the system’s mapping/assignment framework. *Id.*, [0150]. Thus, in the Khandekar-Vasilevsky System, when the user (or a management interface acting on the user’s

behalf) requests provisioning of virtual storage, the system would present and the user would receive an inventory of available virtual storage resources (e.g., available virtual storage entities/logical storage units available for assignment) to support the requested provisioning. MS-1003, ¶¶180-181.

D. GROUND 1E

1. Khandekar-Fujino

(a) Fujino

Fujino describes a Storage Area Network (SAN) in which host-side computers communicate with a storage control device and underlying storage through physical SAN infrastructure (e.g., fiber channel switches and endpoint ports), and in which the system maintains a software-defined representation of that connectivity (e.g., SAN configuration information). MS-1008, 5:31-45, 6:27-36, 6:40-49. “The SAN 19 may be constituted by a plurality of fiber channel switches (FC-SW hereafter) 71,” and the storage control device includes “one or more” physical “ports” for connecting to the SAN. MS-1008, 6:27-36, 6:40-49.

Fujino describes maintaining SAN configuration information that identifies the physical devices along the communication path between endpoints, including “an ID for the port of the storage control device 27, and one or more IDs corresponding respectively to the one or more FC-SWs existing between these ports.” MS-1008, 7:35-43, 10:16-30. Fujino thereby provides an example in

which network connectivity is represented in software terms by identifying the underlying physical SAN devices (ports, switches) that implement the communication path. MS-1008, 10:16-30; MS-1003, ¶¶85-90.

(b) Combination

The Khandekar-Fujino System would have predictably combined Khandekar's VM provisioning environment with Fujino's Storage Area Network (SAN) architecture in which storage network connectivity is represented in software (e.g., as SAN configuration information) by identifying the physical devices along a communication path (e.g., fiber channel switches and endpoint ports). MS-1003, ¶¶104-108.

A POSITA implementing Khandekar's provisioning workflow would have been motivated to incorporate Fujino's SAN connectivity management techniques to support implementations in which provisioned VMs rely on shared storage accessible over a storage network (e.g., a SAN). MS-1003, ¶¶104-108. Khandekar contemplates such implementations through disclosure of a configuration in which a "shared storage area" is accessible by "a staging server as well as the host servers where the VMs are deployed." MS-1004, 16:64-17:13. Khandekar thereby suggests VM data may be maintained in shared storage accessible by multiple machines participating in provisioning and deployment. *Id.*, 16:64-17:13; MS-1004, 9:26-32, 10:50-60. A POSITA would have therefore perceived Fujino's

SAN architecture to be pertinent, since Fujino provides an implementation for network-accessible shared storage in which connectivity between endpoint devices is represented and maintained as configuration information. MS-1008, 5:31-45, 6:27-36, 10:16-30; MS-1003, ¶¶104-108. A POSITA would have recognized these teachings as complementing Khandekar's VM deployments by supplying a way to define and track the storage network connectivity used to reach shared storage from multiple computers/servers. *Id.*

In the combined system, this SAN configuration information would be maintained as part of the provisioning server environment's inventory/state information (e.g., within, or in conjunction with, Khandekar's database 890), so that the system defines and tracks storage connectivity for deployments relying on a shared storage implemented via a storage network (e.g., SAN). MS-1004, 9:26-32, 10:50-60; MS-1008, 10:16-30; MS-1003, ¶¶104-108. This would have enabled Khandekar's provisioned VMs (deployed on selected hosts) to utilize a software-defined representation of SAN connectivity that specifies the underlying physical ports and fiber channel switches implementing the storage communication path used for storage access. MS-1008, 6:27-36, 6:40-49, 10:16-30; MS-1003, ¶¶104-108. In this manner, the combined system would provide a new mechanism defining and tracking storage within the provisioning server's inventory for use in coordinating VM deployments. MS-1004, 18:2-5, 20:54-60; MS-1003, ¶¶104-108.

A POSITA would have reasonably expected to succeed in combining the teachings of Khandekar, and Fujino because the combination incorporates Fujino's SAN connectivity representation (as configuration information identifying physical ports and switches) as inventory/state information used for VM provisioning, yielding predictable results in how storage network connectivity is specified and maintained. MS-1003, ¶¶104-108.

2. Analysis

(a) Claim 9

The Khandekar-Fujino System would have provided a provisioned VM (*virtual asset*) configured to utilize a software-defined representation of network connectivity (e.g., SAN configuration information) (*logical network resource*) that specifies connectivity between host computers and storage systems by identifying physical devices along the communication path (e.g., fiber channel switches and endpoint ports). MS-1004, 9:26-32, 10:50-60; MS-1008, 6:27-36, 10:16-30; MS-1003, ¶¶182-184.

As noted in [1.1]-[1.5], Khandekar describes provisioning and providing user access to VMs (e.g., VMx, VMy) deployed on host servers (e.g., 1000, 1001) under control of a provisioning server. MS-1004, 9:26-32, 10:50-60. In the Khandekar-Fujino System, the combination would have further incorporated Fujino's Storage Area Network (SAN) architecture to provide network

connectivity between these host servers and underlying storage devices. MS-1008, 5:31-45.

Fujino describes that “[t]he SAN 19 may be constituted by a plurality of fiber channel switches (FC-SW hereafter) 71,” which are physical devices on the network used to route communications. *Id.*, 6:27-36. Maintaining SAN configuration information includes, for example, “an ID for the port of the storage control device 27, and one or more IDs corresponding respectively to the one or more FC-SWs existing between these ports.” MS-1008, 10:16-30. A POSITA would have understood and/or found it obvious that this SAN configuration information is a logical (software-defined) abstraction that identifies underlying physical network devices (ports, switches) to define the communication path used by the system, and thus, represents an example of a **logical network resource**. MS-1008, 10:16-30; MS-1003, ¶¶182-184.

(b) Claim 10

The Khandekar-Fujino System would have provided a software-defined network path representation (e.g., maintained in the SAN configuration information) (**logical network unit**) that is associated with (**mapp[ed]**) to the underlying network hardware (**physical network device**), such as the physical ports of the storage control device and the intervening fiber channel switch hardware used to implement the communication path. MS-1008, 6:27-36, 6:40-49, 7:35-43,

10:16-30; MS-1003, ¶¶185-186.

As noted in [9], Fujino describes maintaining SAN configuration information that defines connectivity between endpoints by identifying physical devices along the path, including “an ID for the port of the storage control device 27, and one or more IDs corresponding respectively to the one or more FC-SWs existing between these ports.” MS-1008, 10:16-30, 7:35-43. Fujino describes that the storage control device includes “one or more” physical “ports” for connecting to the SAN, which are tangible network hardware devices. MS-1008, 6:40-49. The SAN configuration information is an example of the *logical network unit* because it represents network connectivity as a software-defined abstraction that identifies (and thereby maps to) underlying physical network devices (ports, switches). For instance, the information specifies identifiers that correspond to (and thereby specify) the particular physical port(s) and fiber channel switch(es) that implement the communication path. MS-1008, 6:27-36, 6:40-49, 10:16-30; MS-1003, ¶¶185-186.

E. GROUND 2A

1. Hirschfeld-Smith

(a) Hirschfeld

Hirschfeld is directed to a “virtualized logical server cloud” in which users access logical servers through a portal-like user interface without needing to know

the access information of the underlying physical servers. MS-1006, 4:37-52, 6:64-7:30, 17:25-39. Although virtualization allows “multiple logical servers to override a physical server’s attributes and share its resources” (*id.*, 2:1-7) users historically “had to be aware of and have specific information about the underlying physical server to access the logical servers” (*id.*, 2:8-35), such as the physical server’s IP address/server name and port number, and incorrect access inputs would have resulted in access being denied. MS-1006, 4:37-52, 6:64-7:30, 17:25-39.

Hirschfeld identifies a desire “to completely separate logical servers from physical servers so that there is no permanent tie between physical and logical resources,” enabling physical servers to function as a resource pool that supports logical servers and allowing a logical server to be reallocated to a different physical server “without users experiencing any change in access approach.” MS-1006, 2:50-59.

Consistent with this objective, Hirschfeld describes a server cloud manager (“SCM”) that provides a “web page portal or other access mechanism” for remote terminals to communicate with the logical server cloud, such that “a user of any remote terminal” may “login to and access any of the logical servers” via the user interface and SCM. MS-1006, 6:28-63. A user may employ a “local browser to access a web page” associated with the user interface, and the GUI provides a

“portal or other interface” through which the user may access one or more logical servers based on permission (e.g., by “point and click” selection of a logical server graphic). *Id.*, 6:64-7:30.

Hirschfeld describes a user database that “incorporates a mapping of users, the logical servers and the physical servers,” including “link information that maps each logical server” with a corresponding physical server, such that the SCM “automatically accesses the appropriate physical server corresponding to the selected logical server,” and the user “does not have to supply physical server information” and “does not even need to be aware of” physical-server access/location information. MS-1006, 7:32-62. Hirschfeld emphasizes that its virtualization “enables a logical Server to exist independent of a specific physical server,” with logical attributes “non-deterministically allocated to physical resources,” thereby creating a cloud of logical servers over one or more physical servers. *Id.*, 17:9-24; MS-1003, ¶¶64-73.

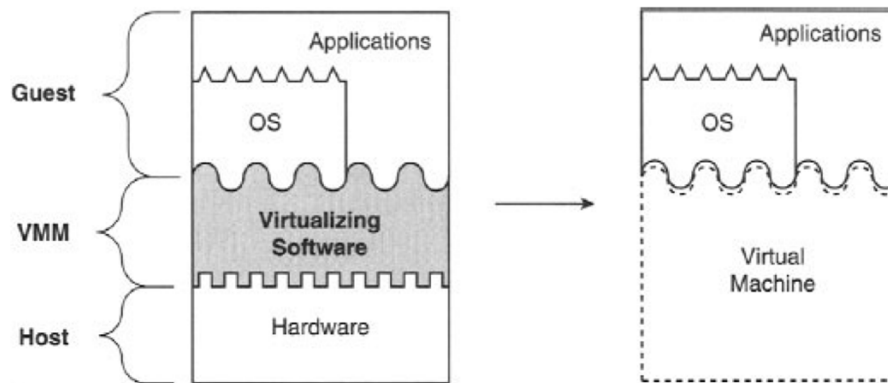
(b) Smith

Smith explains “[v]irtual machine (VM) technologies” “enable new capabilities and to solve a variety of problems in interfacing major computer system components.” MS-1010, xi.

Smith defines a system as “a full execution environment that can simultaneously support a number of processes potentially belonging to different users,” and

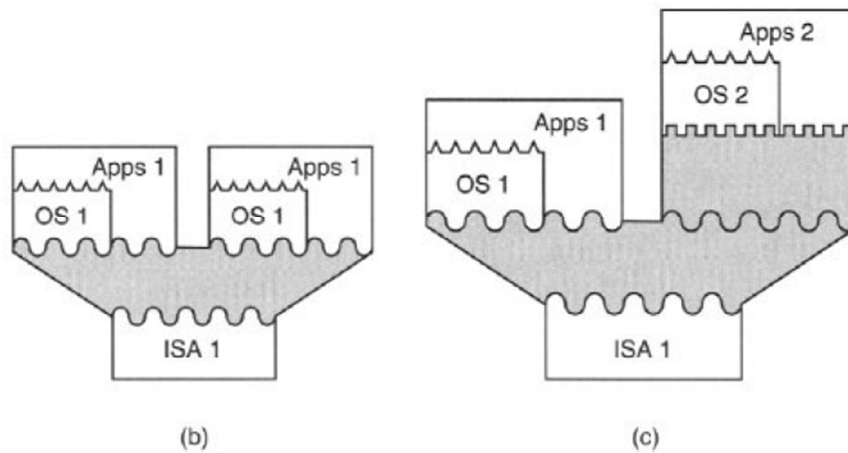
explains that with virtualization, “the real system is transformed so that it appears to be a different, virtual system or even a set of multiple virtual systems.” MS-1010, 3, 9. Smith further explains that “[i]n a system VM environment, real resources of the host platform are shared among the guest system VMs, with a layer of software, the virtual machine monitor (VMM), managing the allocation of, and access to, the hardware resources of the host platform.” MS-1010, 369. “The VMM owns the real system resources and makes them available to one or more guest operating systems that alternately execute on the same hardware,” and “a guest operating system in a virtual machine environment is given the illusion of owning the resources it subsequently allocates to its various user programs.” MS-1010, 369.

In Smith, “[a] virtual machine (VM) is implemented by adding a layer of software to a real machine to support the desired virtual machine’s architecture.” MS-1010, 4. Such “virtualizing software is often referred to as the virtual machine monitor (VMM).” MS-1010, 11.



MS-1010, FIG. 1.7

“Multiple, replicated virtual machines can be implemented on a single hardware platform to provide individuals or user groups with their own operating system environments.” MS-1010, 5. In Figure 1.8(b), Smith illustrates an example in which a set of applications (Apps 1) are running on the same operating system (e.g., where both instances of the operating system “OS1” are Linux or both are Windows) in two different VMs, and in Figure 1.8(b), Smith illustrates two different sets of applications (Apps 1 and Apps 2) are running on different operating systems (e.g., “OS1” is Windows and “OS2” is Linux). MS-1010, Figures 1.8 (b)-(c).



MS-1010, Figures 1.8 (b)-(c)

Smith explains that a “system VM provides a sandbox that isolates one system environment from other environments, thus ensuring a level of security that may not be provided by a single operating system.” MS-1010, 370-371. Further, with VMs, “different system environments (possibly with different operating systems) also provide isolation and enhanced security.” MS-1010, 5. Smith notes that “[p]erhaps the most important feature of today’s system VMs is that they provide a secure way of partitioning major software systems that run concurrently on the same hardware platform” such that “[s]oftware running on one guest system is isolated from software running on other guest systems.” MS-1010, 18. Smith further notes that “a virtual machine can be used to provide an environment that, for all practical purposes, is isolated from the others and through which it is not possible for one user to observe or change another’s data and activity.” MS-1010, 370-

371.

Smith further explains that “[r]unning each application in its own virtual machine increases the robustness of the system,” since erratic behavior “on a virtual machine” is “less likely to affect the operation of another application running on a different virtual machine.” MS-1010, 370. Further, “if security on one guest system is compromised or if the guest OS suffers a failure, the software running on other guest systems is not affected.” MS-1010, 18; MS-1003, ¶¶91-94.

(c) Combination

The Hirschfeld-Smith System would have predictably combined Hirschfeld’s virtualization environment with Smith’s well-known teachings regarding implementing virtualization software with a VMM to manage multiple virtual machines on a host platform. MS-1003, ¶¶109-115. Hirschfeld identifies configurations in which a physical server supports multiple logical servers (e.g., as shown in Figures 2 and 3). For example, “logical servers 106 form[] a logical server cloud 201,” and more specifically, “the logical servers LS 1-4 are linked to the [physical server (PS) A], the logical servers LS 5-8 are linked to the PS B, and the logical servers LS 9-N are linked to the PS C.” MS-1006, 7:63-8:10. In such configurations, each logical server executes its own operating system, which may be “any desired OS,” such “those from Microsoft or based on UNIX, Linux, etc.” *Id.*, 10:9-31.

And yet, in these configurations, Hirschfeld leaves open the precise software mechanism for managing multiple logical servers on a single physical server as an implementation detail. MS-1006, 9:47-57; MS-1003, ¶¶109-115. A POSITA would therefore have looked to elements of similar virtualization environments known by the Critical Date to provide such functionality, such as a virtual machine monitor (VMM). *Id.* For example, Smith describes that “[i]n a system VM environment, real resources of the host platform are shared among the guest system VMs, with a layer of software, the virtual machine monitor (VMM), managing the allocation of, and access to, the hardware resources of the host platform.” MS-1010, 369. Smith describes that “[a] virtual machine (VM) is implemented by adding a layer of software to a real machine to support the desired virtual machine’s architecture,” and that such “virtualizing software is often referred to as the virtual machine monitor (VMM).” MS-1010, 4, 11; MS-1003, ¶¶109-115.

A POSITA implementing the configurations described in Hirschfeld would have been motivated to incorporate Smith’s VMM teachings because Hirschfeld’s architecture presupposes multiple independent logical server operating systems running on the same physical server hardware. MS-1003, ¶¶109-115. Smith’s disclosures regarding a VMM thereby provide a known and compatible mechanism for implementing Hirschfeld’s contemplated configurations, e.g., by mediating ac-

cess to and allocation of the host's hardware resources among multiple guest operating systems. MS-1006, 8:1-10, 10:9-31; MS-1010, 369; MS-1003, ¶¶109-115.

As combined, the Hirschfeld-Smith System would have provided a virtualized logical server cloud infrastructure (per Hirschfeld). MS-1006, FIG. 1; §III.A.1; MS-1003, ¶¶109-115. Within this infrastructure, physical servers include components described within Hirschfeld. MS-1006, 8:39-64; MS-1003, ¶¶109-115. Further, a Smith-like VMM manages multiple logical operating systems run on a physical server to support logical servers to support configurations contemplated within Hirschfeld. MS-1003, ¶¶109-115. The Smith-like VMM would have provided functionality similar to those described in Smith, such as “manag[ing] the allocation of, and access to, the hardware resources of the host platform.” MS-1010, 369. The Smith-like VMM would have enabled the SCM of the Hirschfeld-Smith System to “own[] the real system resources and make[] them available to one or more guest operating systems that alternately execute on the same hardware” such that “a guest operating system in a virtual machine environment is given the illusion of owning the resources it subsequently allocates to its various user programs.” MS-1010, 369.

Finally, a POSITA would have reasonably expected to succeed in implementing the teachings of Hirschfeld and Smith in the manner discussed above.

MS-1003, ¶¶109-115. This is because the combination of teachings involves applying a well-known software elements of virtualization systems (e.g., Smith’s VMM) within a well-known virtualization environment (e.g., Hirschfeld’s SCM-based control of logical servers provided on one or more physical servers) such that the combination involves a predictable solution to supporting configurations in which a physical server supports multiple logical servers (e.g., as shown in Hirschfeld’s Figures 2 and 3). MS-1006, 7:63-8:10, 10:9-31, FIGS. 2, 3; MS-1003, ¶¶109-115. A POSITA would have thereby understood this as combining known elements according to known methods to yield predictable results. *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 415-21 (2007); MS-1003, ¶¶109-115.

2. Analysis

(a) Claim 1

[1.pre]

To the extent the preamble is deemed limiting, the Hirschfeld-Smith System would have satisfied the requirements of [1.pre]. MS-1003, ¶¶187-188. Hirschfeld describes a “method of virtualizing a plurality of logical servers into a logical server cloud linked to a plurality of physical servers.” MS-1006, 4:54-5:3. Additionally, “a user of any remote terminal 105 may login to and access any of the logical servers 106 of the virtualized logical server cloud 101 via the user interface 104 and SCM 102 as though logging into a server directly or via a local

area network (LAN).” MS-1006, 6:54-63; MS-1003, ¶¶187-188. The Hirschfeld-Smith System would have thereby enabled access to one or more logical server resources (e.g., resources of logical servers 106 of the virtualized logical server cloud) (*one or more virtualization system resources*) over an intermediate network (e.g., 103) (*network*). MS-1006, 6:28-42; MS-1003, ¶¶187-188.

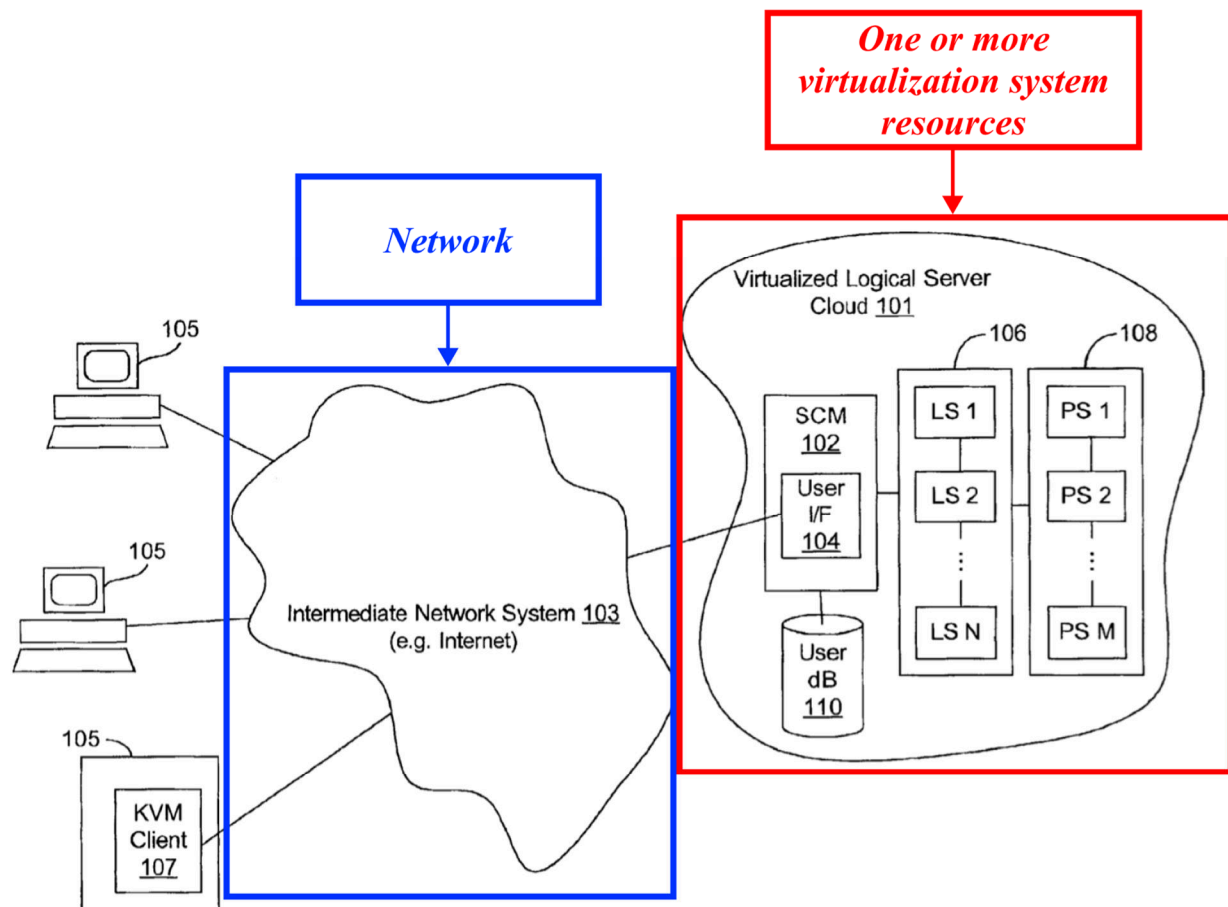


FIG. 1

MS-1006, FIG. 1.

[1.1]

The Hirschfeld-Smith System (*virtualized system*) would have provided a SCM (*control computer*) that interfaces with and controls a bank of physical servers (*one or more physical resources*) hosting logical servers (*virtual assets*), where the physical resources and virtual assets are connected to the SCM. MS-1006, 5:66-6:27, 6:34-7:5, 7:24-30; MS-1003, ¶¶189-193.

As noted in [1.pre], in Hirschfeld, a “virtualized logical server cloud” includes an SCM representing “cloud management infrastructure.” MS-1006, 5:66-6:27. The SCM “has the authority to allocate physical resources to maintain the logical server cloud” and thereby defines “physical resources under management control.” *Id.*, 6:7-15. Moreover, “the physical servers controlled by the SCM determine a logical server cloud’s boundary.” *Id.* Thus, the SCM is an example of a *control computer* in that it controls resource allocation for the virtualized logical server cloud. MS-1003, ¶¶189-193.

Further, the virtualized logical server cloud includes “one or more logical servers 106” (*virtual assets*). MS-1006, 6:34-38. The SCM is *connected to* the one or more logical servers because, for example, the “SCM 102 provides KVM redirection either by providing a KVM interface or by mapping the KVM client 107 to allow direct KVM access to the selected logical server 106.” MS-1006, 6:64-7:5; MS-1003, ¶¶189-193.

The virtualized logical server cloud includes a bank of physical servers 108”
(*one or more physical resources*). MS-1006, 6:38-42. The SCM is *connected to*
the bank of physical servers because, for example, the user interface associated
with the SCM “also provides interfaces to provide SCM control of the physical
server to enable other functions including copy, move, start, shutdown, reset and
any other functions enabled by control over the physical server and virtualization
software.” MS-1006, 7:24-30. Thus, in the Hirschfeld-Smith System, the SCM
(*control computer*) is *connected to* and controls each of the bank of physical
servers (*physical resources*) and the logical servers (*virtual assets*) instantiated on
those physical servers. *Id.*, 6:48-54; MS-1003, ¶¶189-193.

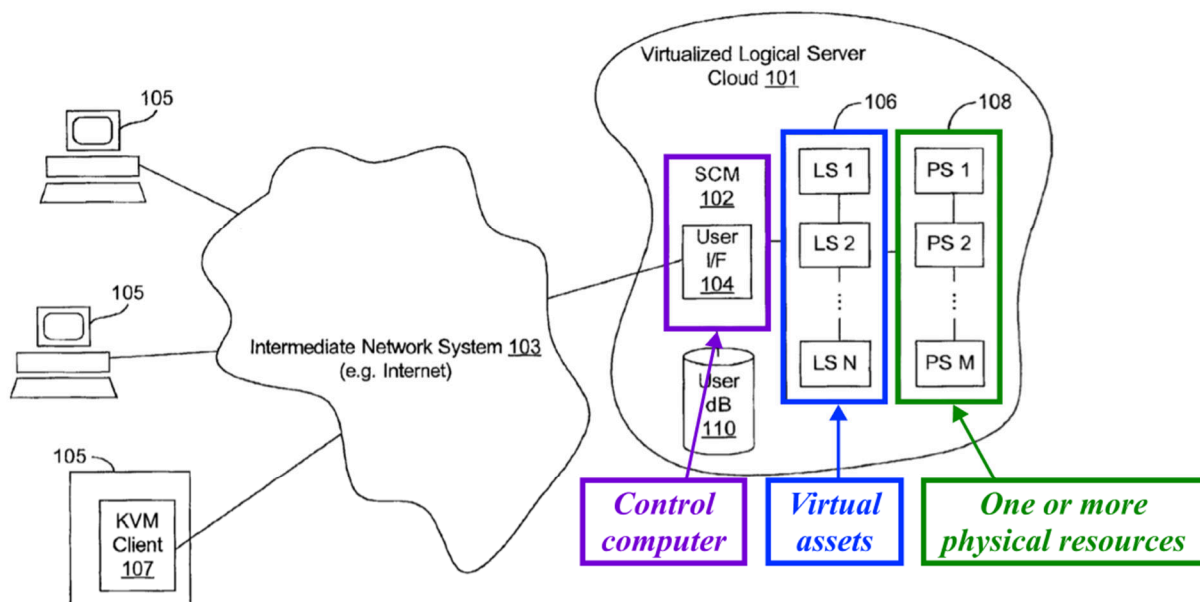


FIG. 1

MS-1006, FIG. 1.

[1.2]

The Hirschfeld-Smith System would have provided physical servers (*physical resources*) that each execute multiple logical servers (*at least two virtual assets*), where the logical servers are provided using virtualization technology and therefore are examples of *software instantiations of computer systems*. MS-1006, 1:63-2:7, 6:34-46, 10:22-27, FIGS. 2, 3; MS-1003, ¶¶194-199.

As noted in [1.1], in Hirschfeld, the virtualized logical server cloud includes “one or more logical servers 106” and a “bank of physical servers 108,” where the number of physical servers (“M”) “may be a smaller number” than the number of logical servers (“N”). MS-1006, 6:28-63, 8:1-10. Thus, when “M” is smaller than “N,” at least one physical server executes multiple logical servers, i.e., *at least two virtual assets*. MS-1003, ¶¶194-199.

Moreover, physical servers may execute multiple logical servers. For example, in Hirschfeld’s Figure 2, a “logical server cloud” is formed with multiple “logical servers 106” and “bank of physical servers,” e.g., “logical servers LS 1-4 are linked to the PS A, the logical servers LS 5-8 are linked to the PS B, and the logical servers LS 9-N are linked to the PS C.” MS-1006, 6:28-63.

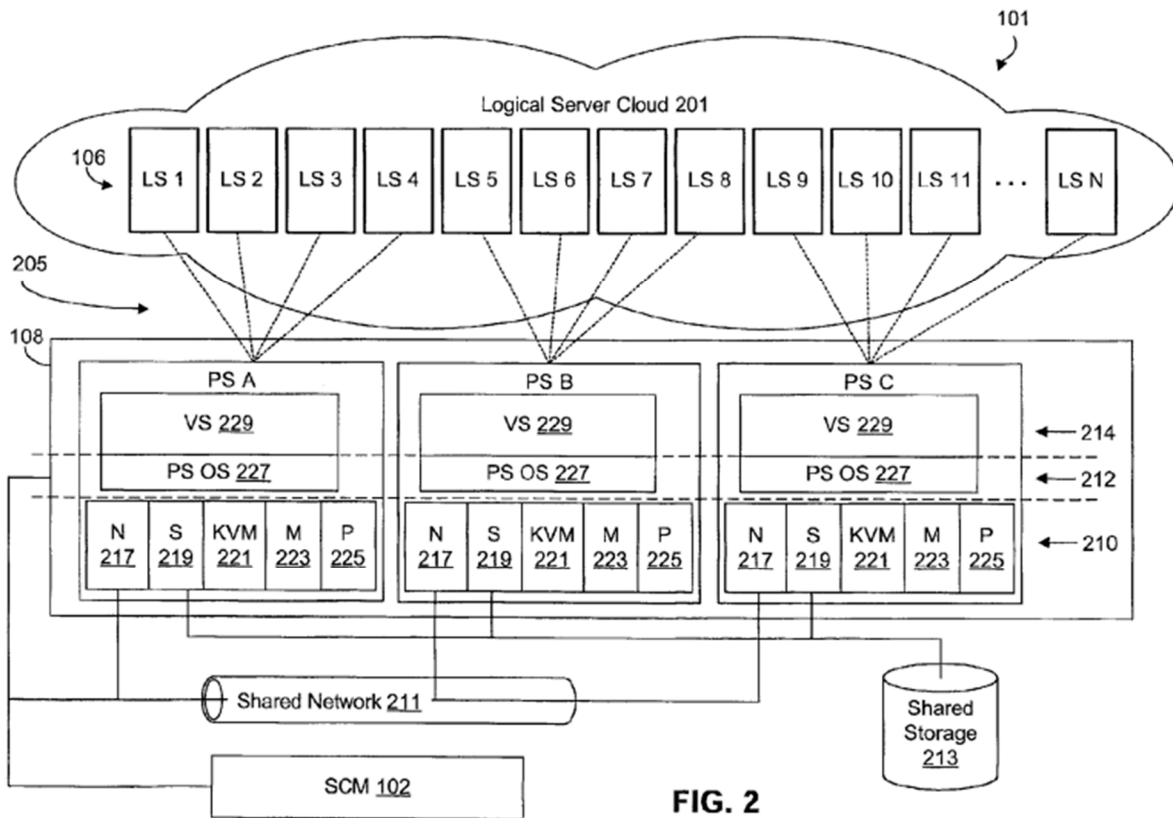
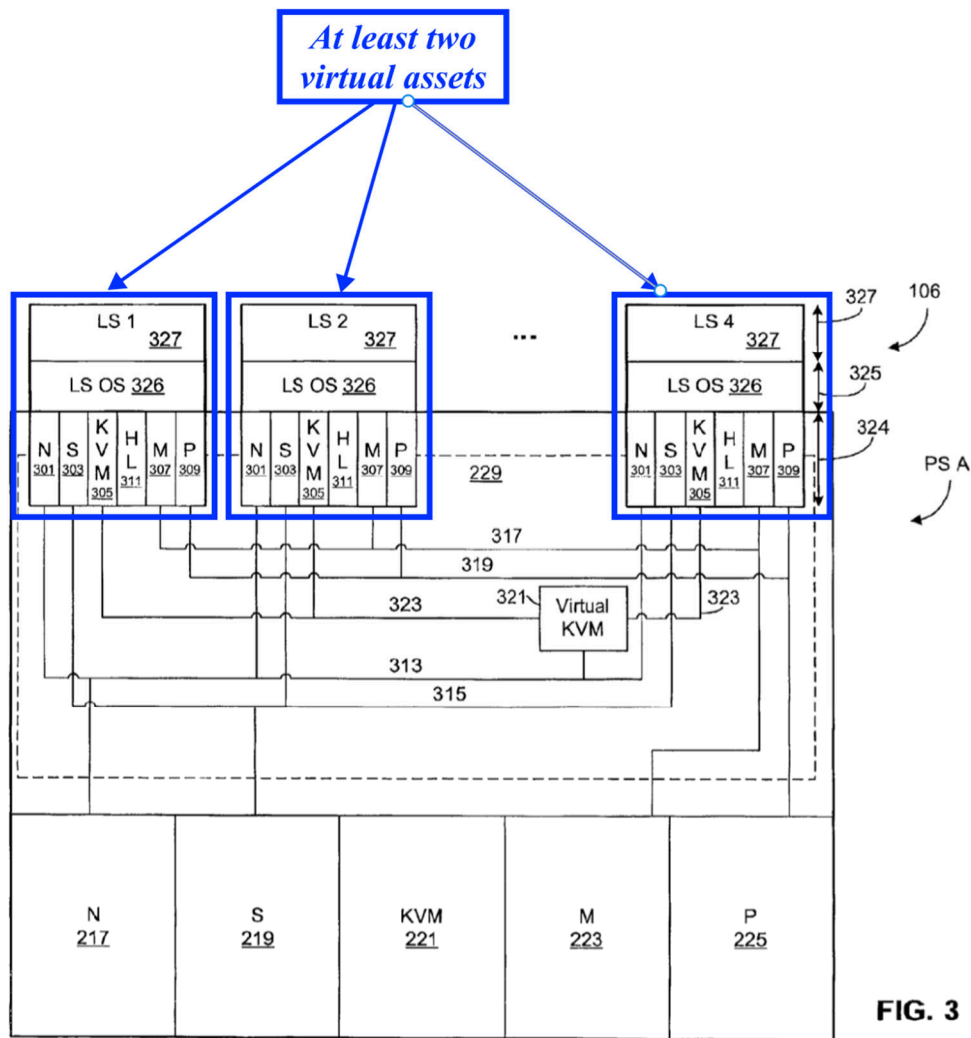


FIG. 2

MS-1006, FIG. 2

Hirsfield's Figure 3 portrays another example of multiple logical servers running on a single physical server. In this configuration, each logical server operating system "is executed independently from every other LS OS 326, or otherwise gives every appearance of independent operation from the user's perspective." MS-1006, 10:9-31, FIG. 3.



MS-1006, FIG. 3.

More broadly, Hirschfeld describes that virtualization “allow[s] multiple logical servers to override a physical server’s attributes and share its resources.” *Id.*, 2:1-7. In this way, a POSITA would have implemented the Hirschfeld-Smith System to leverage Hirschfeld’s techniques regarding consolidating logical servers over shared physical resources under centralized management such that its hosting servers (*one or more physical resources*) execute *at least two virtual assets*, e.g.,

physical server PS A running logical servers LS 1-4 and physical server PS B running logical servers LS 5-8. MS-1006, 8:1-10; MS-1003, ¶¶194-199.

Hirschfeld's logical servers (e.g., 106) are examples of *software instantiations of computer systems*. MS-1003, ¶¶194-199. Hirschfeld describes “the use of virtualization technology that enables multiple logical servers to operate on a single physical computer,” and further notes that virtualization allows multiple logical servers to “override a physical server's attributes and share its resources.” MS-1006, 1:63-2:7. Thus, in the Hirschfeld-Smith System, each physical server (*physical resource*) executes at least two logical servers (*virtual assets*) leverage virtualization technology and thereby are examples of *software instantiations of computer systems*. *Id.*, 1:63-2:7, 8:1-10; MS-1003, ¶¶194-199.

[1.3]

The Hirschfeld-Smith System would have provided physical servers (*physical resources*) that each execute virtualization software (e.g., Virtual machine monitor (VMM)) (*virtualization layer instances*) to manage logical servers (*virtual assets*) linked to and executing on that physical server. MS-1006, 9:35-57; MS-1010, 369; MS-1003, ¶¶200-203.

As noted in [1.1] and [1.2], in Hirschfeld, each physical server in the bank of physical servers executes virtualization software. For example, in Figure 2, “[e]ach physical server 108 executes virtualization software 229,” where

“virtualization software 229 manages physical resources of the logical servers 106 linked thereto.” MS-1006, 9:35-57. In this configuration, the virtualization software 229 running on a given physical server is an example of a **virtualization layer instance** running on that physical resource that manages the corresponding logical servers (**virtual assets**). *Id.*; MS-1003, ¶¶200-203.

Further, in the Hirschfeld-Smith System, a POSITA would have implemented Hirschfeld’s virtualization software 229 to manage logical operating systems using a well-known mechanism, e.g., using a Smith-like VMM based on Smith’s disclosure. MS-1003, ¶¶200-203. In Smith, a VMM “manag[es] the allocation of, and access to, the hardware resources of the host platform,” that “[t]he VMM owns the real system resources and makes them available to one or more guest operating systems that alternately execute on the same hardware,” and that “a guest operating system in a virtual machine environment is given the illusion of owning the resources it subsequently allocates to its various user programs.” MS-1010, 369. Thus, in the Hirschfeld-Smith System, physical servers runs VMMs (**virtualization layer instances**) that manage logical servers (**virtual assets**) by managing allocation of and access to the physical server’s hardware resources. MS-1010, 369; MS-1006, 9:35-57; MS-1003, ¶¶200-203.

[1.4]

The Hirschfeld-Smith System would have connected a user (e.g., **end user**)

to the SCM (*control computer*) through a user interface (e.g., user interface 104), thereby providing a networked connection to the SCM for logging in and interacting with the virtualized logical server cloud. MS-1006, 6:28-63; MS-1003, ¶¶204-205.

As noted in [1.pre], in Hirschfeld, a “user of any remote terminal” may “login to and access” logical servers via a user interface of SCM as “though logging into a server directly or via a local area network (LAN).” MS-1006, 6:28-63. Because the user logs in and accesses the logical servers via the SCM (*control computer*), this establishes a connection between the end user and the SCM. MS-1003, ¶¶204-205. Thus, the Hirschfeld-Smith System would have *connect[ed]* the user (*end user*) to the SCM (*control computer*). *Id.*

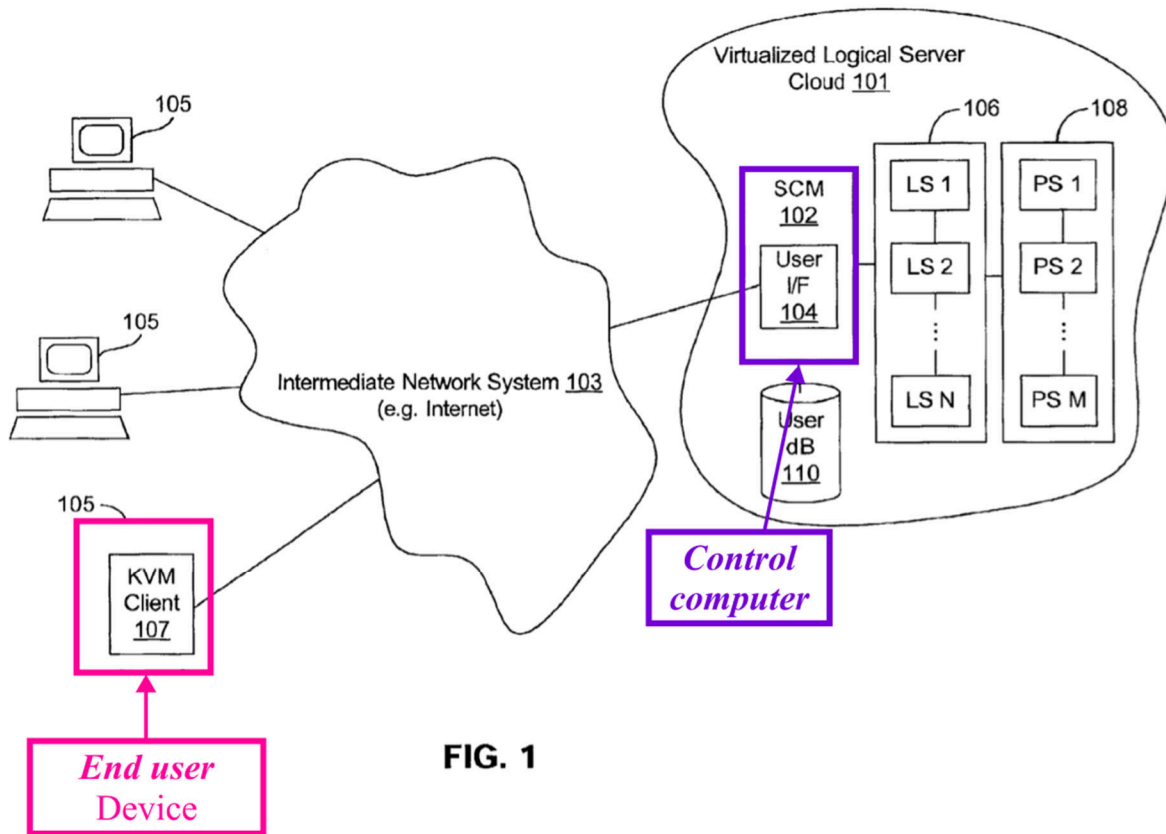


FIG. 1

MS-1006, FIG. 1

[1.5]

The Hirschfeld-Smith System would have enabled the user to access the logical servers (*virtual assets*) implemented in the virtualized logical server cloud through the user's connection to the SCM (*control computer*) via a user interface. MS-1006, 6:28-63, 14:39-61; MS-1003, ¶¶206-209.

As noted in [1.4], in Hirschfeld, a user “may login to and access any of the logical servers” via a user interface of the SCM, e.g., “as though logging into a server directly or via a local area network (LAN).” MS-1006, 6:28-63; MS-1003,

¶¶206-209. Thus, in the Hirschfeld-Smith System, the *access/es/* the logical servers (*virtual assets*) through the connection to the SCM (*control computer connection*), e.g., by logging in via the SCM's user interface and using that connection to access the logical servers for which the user has permission. *Id.* For example, to access a logical server, Hirschfeld's proxy architecture may then be leveraged to route remote access directly through the SCM. MS-1003, ¶¶206-209. This may be enabled by, for instance, Hirschfeld's implementation the SCM 102 providing Keyboard/Video/Mouse (KVM) redirection such that "KVM requests to and from [logical server] are handled via the KVM client 519 of the SCM 102." MS-1006, 14:39-61.

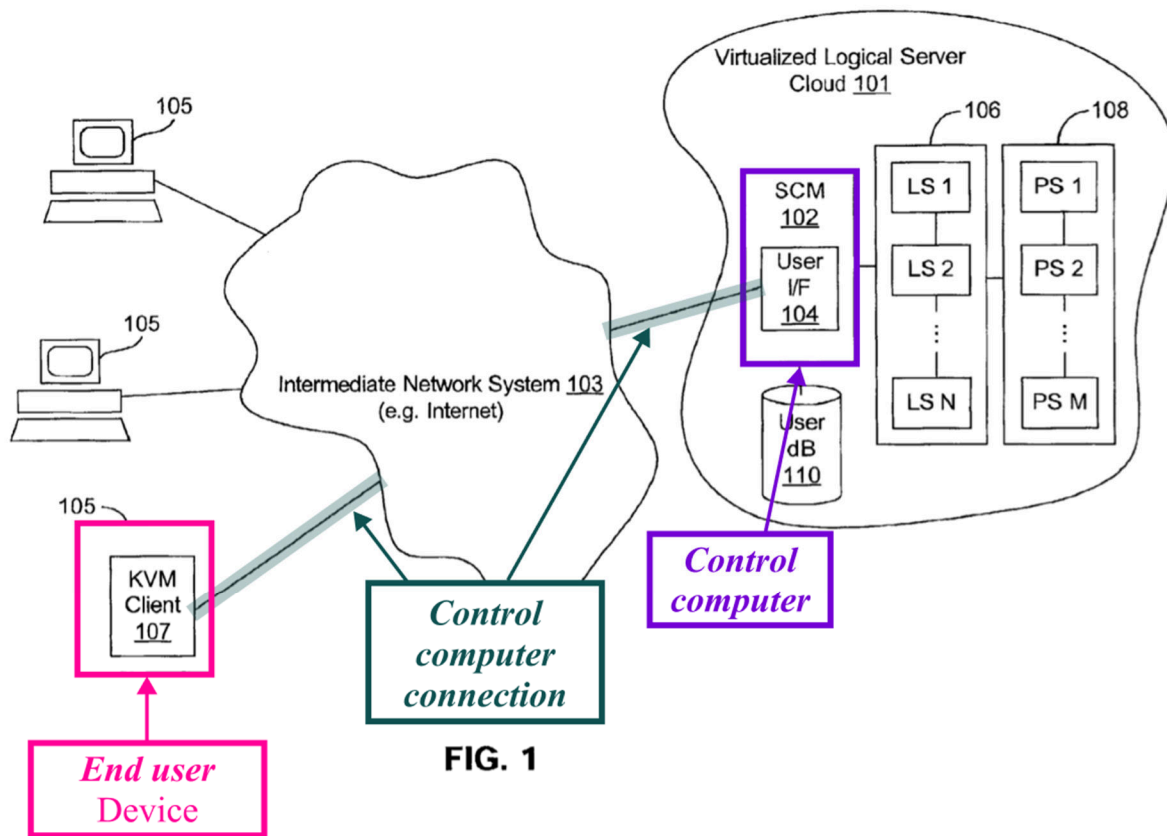


FIG. 1

MS-1006, FIG. 1

(b) Claim 2

The Hirschfeld-Smith System would have connected a user (*end user*) to the SCM (*control computer*) through a network, including implementations where the network is the Internet. MS-1006, 6:28-63; MS-1003, ¶¶206-209. In Hirschfeld, the virtualized logical server cloud is “accessible to one or more remote terminals” ... via an “intermediate network system,” such as “the globally-accessible Internet, although any type of network is contemplated.” *Id.* The SCM includes a user interface that provides a web page portal to facilitate communication with remote

terminals via that network system. *Id*; MS-1003, ¶¶206-209.

F. GROUND 2B

1. Hirschfeld-Smith-Bandhole

The Hirschfeld-Smith-Bandhole System would have predictably combined Hirschfeld’s virtualization environment with the examples of well-known user device types for accessing remotely managed virtualized resources. MS-1003, ¶116; §III.C.1(b). In Hirschfeld’s virtualization environment, users access logical servers under control of a SCM via “terminals,” which may be implemented as “simple terminals,” “thin client computers,” and “low end PCs.” MS-1006, 6:28-63. As discussed in Ground 1C, Bandhole confirms that such client-side access was well-known by the Critical Date since computing devices used in this manner “can be made in many dimensions and styles,” including “workstation,” “laptop,” and “palmtop” form factors (including PDA-type devices). MS-1007, [0035], [0040]; MS-1003, ¶¶116-120.

A POSITA implementing Hirschfeld’s terminal-based user access to the SCM and logical servers would have been motivated to enable use of the terminals from these known client device types because Hirschfeld’s approach presupposes remote client communication over a network and expressly contemplates modest client endpoints (e.g., thin clients and low end PCs) rather than specialized hardware. MS-1006, 6:28-63; MS-1003, ¶¶116-120. In this context, Bandhole’s

teachings provide relevant and compatible examples confirming that remote interaction with server-side computing environments may be performed from different user device form factors (e.g., workstation, laptop, palmtop/PDA). MS-1007, [0029], [0035], [0040], [0048]; MS-1003, ¶¶116-120.

A POSITA would have further recognized that implementing Hirschfeld's terminal-based SCM access on these known device types is complementary to (and supportive of) Hirschfeld's design choice to use thin/low-end client endpoints, without requiring changes to Hirschfeld's underlying virtualization and control sequence for managing logical servers on physical resources. MS-1006, 6:28-7:30; MS-1003, ¶¶116-120. In this regard, a POSITA would have reasonably expected success because the combination involves applying known client device form factors (e.g., workstation/desktop, laptop, palmtop/PDA) as predictable implementations of Hirschfeld's expressly contemplated "low end PC"/thin-client terminals used to access the SCM and logical servers. *Id.*

2. Analysis

(a) Claim 3

The Hirschfeld-Smith-Bandhole System would have permitted a user to access logical servers (*virtual assets*) through a connection established using a user interface of the SCM (*control computer*) (*connecting the end user to the control computer*). MS-1006, 6:28-63, 7:32-48; MS-1003, ¶¶212-214. A POSITA would

have found it obvious that the user device for accessing the logical servers may be one of several well-known device types, including a “workstation” (*desktop, client*), “laptop” (*laptop*), and “palmtop” (*personal digital assistant (PDA)*), and that such devices would include client-side resources used to communicate over the network. MS-1007, [0035]; MS-1003, ¶¶212-214.

As noted in [1.5], the Hirschfeld-Smith System would have enabled access to logical servers through a connection provided by the SCM (*control computer*), e.g., via the SCM’s user interface through which the user logs in and accesses logical servers for which the user has permission. MS-1006, 6:28-63, 7:32-48; MS-1003, ¶¶212-214. A POSITA would have understood and/or found it obvious that the device used to establish that connection may be one of several known user device types that function as client devices in networked systems, and would necessarily include client-side resources (e.g., client computing and network interface resources) to support such access. MS-1003, ¶¶212-214. Indeed, Hirschfeld describes that user devices used to access logical servers (e.g., terminal 105) “may be configured with various levels of capability, including simple terminals, thin client computers, low end PCs, etc.” MS-1006, 6:28-63.

Bandhole similarly demonstrates that it was well-known by the Critical Date to use various types of user device types to access and manage virtual computing environments through remotely accessible interfaces. MS-1007, [0015], [0035],

[0048]. In Bandhole, such user device types are described to be made in many “dimensions and styles,” including “workstation,” “laptop,” and “palmtop” form factors. *Id.*, [0035].

(b) Claim 4

As noted in [2], the Hirschfeld-Smith-Bandhole System would have permitted a user to access logical servers through a connection established using a user interface of the SCM (*accessing the virtual assets*). MS-1006, 10:9-31; MS-1003, ¶¶215-217.

A POSITA would have found it obvious based on the teachings of Hirschfeld and Bandhole that accessing the logical servers (*virtual assets*) involves, in some implementations, accessing an application server. MS-1003, ¶¶215-217; MS-1006, 10:9-31. For example, Hirschfeld describes logical servers as virtualized computing systems that can execute software while maintaining virtual independence, and such logical servers would have been configured to perform conventional server roles, including application-serving roles, depending on the deployed environment. *Id.* Bandhole likewise describes multi-tier architectures and expressly discloses “application servers” (e.g., application server 311; application servers 221, 223) that receive requests from web clients (e.g., a browser) and provide results back to the client, demonstrating that application-server functionality was a known and expected component of remotely accessible

computing environments. MS-1007, [0005], [0009], [0044], [0048], [0051]-[0054]. Thus, in some implementations, the Hirschfeld-Smith-Bandhole System would have provided access to logical servers that provide *application server* functionality such that the access involves accessing an application server by the end user. MS-1003, ¶¶215-217.

(c) Claim 5

As noted in [4], the Hirschfeld-Smith-Bandhole System would have enabled a user to access a logical server (*virtual asset*) through a connection to the SCM (*control computer*), and it would have been obvious that in some implementations the accessed logical server provides *application server* functionality because logical servers execute software and may be configured for conventional server roles. MS-1006, 6:28-63, 10:9-31; MS-1003, ¶¶218-222. Bandhole similarly describes that user access to a remotely administered dynamic computing environment can involve application servers that receive requests from web clients (e.g., a browser) and return results, further demonstrating application-server access in such environments. MS-1007, [0044], [0048], [0051]-[0054]; MS-1003, ¶¶218-222.

Bandhole describes a multi-tier architecture including a middleware tier that contains one or more application servers and performs delegation of requests to services responsible for actions such as accessing a database, storage, or computing

devices, which a POSITA would have understood as “middleware server” functionality in at least some implementations. MS-1007, [0042]-[0045]; MS-1003, ¶¶218-222. Bandhole describes database functionality (e.g., a data repository and database access in a three-tier architecture), supporting that in some implementations the virtual asset provides end user access to a database (e.g., through the application server/middleware components that access and interact with the database). MS-1007, [0009], [0051]-[0054]; MS-1003, ¶¶218-222.

G. GROUND 2C

1. Hirschfeld-Smith-Vasilevsky

The Hirschfeld-Smith-Vasilevsky System would have predictably combined Hirschfeld’s virtual environment with Vasilevsky’s well-known techniques for mapping and dynamically allocating virtual resources (including storage) to underlying physical hardware. MS-1003, ¶¶121-125; §III.C.1(c). A POSITA would have been motivated to implement the combination of these teachings given Hirschfeld’s recognition that storage resources “may be divided into multiple physical and logical devices (e.g., File Transfer Protocol (FTP) and Network Attached Storage (NAS) or the like)” and that “[a]ccess to the network and storage resources is **preferably logical** rather than physical so that they act like resource clouds...” MS-1006, 8:66-9:20. This would have prompted a POSITA to implement logical servers to provide virtual storage capabilities (e.g., NAS-like

storage services implemented as logical/virtual resources backed by physical storage). MS-1003, ¶¶121-125.

As previously discussed, Vasilevsky confirms that, by the Critical Date, it was well-known to manage such virtual storage capabilities within virtualized environments. MS-1005, Abst., [0009], [0014], [0074]-[0079], [0150]; MS-1003, ¶¶121-125. A POSITA implementing Hirschfeld's SCM-based control of logical servers (e.g., logically accessed "storage resource clouds") would have been motivated to incorporate Vasilevsky's storage-related teachings to improve transparency and controllability of storage assignment. *Id.*, [0077]-[0083], [0150], [0166]; MS-1003, ¶¶121-125.

Further, Vasilevsky's storage-related teachings (e.g., virtual storage mapping) are complementary to (and supportive of) Hirschfeld's SCM-controlled virtualization paradigm, without requiring changes to the fundamental concept of running isolated logical servers on shared physical infrastructure. MS-1003, ¶¶121-125. In this regard, a POSITA would have reasonably expected success because the combination involves applying well-known virtualization management mechanisms (e.g., mapping databases, management interfaces, dynamic allocation/deallocation of resources) to a similar type of virtualization environment already contemplated within Hirschfeld for implementing and managing logically accessed virtual storage capabilities (i.e., configurations expressly described in

Vasilevsky). MS-1006, 8:66-9:20; MS-1005, [0074]-[0079], [0150]; MS-1003, ¶¶121-125.

2. Analysis

(a) Claim 6

The Hirschfeld-Smith-Vasilevsky System would have provided a logical server (*virtual asset*) that further provides a virtualized storage construct (*virtual storage resource*), such as a virtualized storage resource of the logical server that is virtualized to an underlying physical server storage resource. MS-1006, 8:66-9:20; MS-1003, ¶¶223-225.

In Hirschfeld, each logical server includes a “storage resource” that is virtualized to the corresponding storage resource of the physical server, thereby providing the logical server with storage as a virtualized resource rather than a physical disk device. MS-1006, 8:66-9:20. Further, in Vasilevsky, a virtual resource includes a “virtual storage entity” that is distinct from an “actual storage device” (e.g., Vasilevsky's “disks 914”). MS-1005, [0166], Cl. 80. Thus, in the Hirschfeld-Vasilevsky System, the logical server (*virtual asset*) provides a “virtual storage entity” as a *virtualized storage resource* presented as part of, or to support, the virtualized server instance. MS-1003, ¶¶223-225.

(b) Claim 7

As noted in [6], the Hirschfeld-Smith-Vasilevsky System would have

provided a logical server (*virtual asset*) that includes a virtualized storage construct (*virtual storage resource*), such as a storage resource of the logical server that is virtualized to an underlying physical server storage resource. MS-1006, 10:32-56; MS-1003, ¶¶226-228. Consistent with Vasilevsky, in the combination, the virtual storage construct includes a mapping between (i) a storage abstraction presented logically to a virtualized system (logical storage unit) and (ii) a hardware component (physical storage disk). MS-1005, [0150], [0166], Cls. 78, 84, 85; *see* Ground 1D (addressing [7]); MS-1001, 20:45-21:3; MS-1003, ¶¶226-228.

(c) Claim 8

See [7]; MS-1003, ¶229

(d) Claim 11

[11.pre]

See [1.pre]; MS-1003, ¶230.

[11.1]

See [1.pre]. The Hirschfeld-Smith-Vasilevsky System would have enabled a user to *communicat[e]* over a network (*network*) with the SCM (*control computer*) to access one or more logical servers (*virtual assets*) operating on physical servers in the virtualized computing environment (*virtualization system*). MS-1006, 6:28-63; MS-1003, ¶¶231-233. Vasilevsky similarly describes a virtualized data center in which virtual servers (*virtual assets*) execute on

underlying physical nodes and are managed through a management infrastructure/interface, which a POSITA would have understood as permitting user communication over a network to access such virtual assets running in the virtualization system. MS-1005, Abstract, [0016]-[0017], [0074]-[0079]; MS-1003, ¶¶231-233.

[11.2]

See [1.1]; MS-1003, ¶¶234-236.

[11.3]

See [1.2]; MS-1003, ¶¶237-240.

[11.4]

See [1.3]; MS-1003, ¶¶241-244.

[11.5]

The Hirschfeld-Smith-Vasilevsky System would have received, from a user interacting with the SCM through a user interface (*requesting, by the user*), a request to provision a selected logical server (*virtual asset*) within the virtualized computing environment (*virtualization system*). MS-1006, 4:15-28, 6:28-7:30; MS-1003, ¶¶245-248.

As noted in [1.4] and [1.5], in Hirschfeld, a user connects to and interacts with the SCM (*control computer*) through a user interface to obtain access to logical servers, including by selecting and initiating actions that cause logical

servers to be made available for the user (e.g., by causing a logical server to be created/activated/replicated for use on physical server resources). MS-1006, 4:15-28, 6:28-7:30. In this workflow, the user's request communicated to the SCM is understood as a request to provision a virtual asset (i.e., to provision a logical server instance for the user within the virtualization system). MS-1003, ¶¶245-248.

Vasilevsky further confirms that provisioning of a virtualized computing instance is performed responsive to a user's request through a management interface. MS-1005, [0074]-[0079]. Thus, in the Hirschfeld-Smith-Vasilevsky System, the user may request provisioning of a logical server in the virtualization system. MS-1003, ¶¶245-248.

[11.6]

In the Hirschfeld-Smith-Vasilevsky System, a selected logical server provisioned based on a user provisioning request (*provisioned virtual asset*) may be configured to function as a storage server for the network, and thus, acting as an example of a *virtual storage resource*. MS-1006, 8:66-9:20; MS-1005, [0011]; MS-1003, ¶¶249-251. A POSITA would have understood and/or found it obvious that virtualized computing instances are routinely provisioned to provide storage capabilities because Vasilevsky, for example, describes how specialized storage hardware is replaced by a "software-only application that is executed on a virtual

server,” e.g., by providing “Network Attached Storage (NAS) functions.” MS-1005, [0011]; MS-1003, ¶¶249-251. Thus, in some implementations of the Hirschfeld-Smith-Vasilevsky System, the provisioned logical server functions as an example of *virtual storage*. MS-1003, ¶¶249-251; *see* Ground 1D (addressing [11.6]); MS-1005, [0150], [0166].

(e) Claim 12

The Hirschfeld-Smith-Vasilevsky System would have enabled provisioning operations to be requested through a management interface that includes a programmatic interface (*application programming interface (API) of the virtualization system*). This includes provisioning operations directed to storage resources allocated to a provisioned logical server/virtual server (*provisioning of the virtual storage*). MS-1005, [0014], [0074]-[0079]; MS-1003, ¶¶252-254.

Vasilevsky describes that control of allocating and/or deallocating resources to a virtual computing system may be provided “through an interface (e.g., via a CLI, or GUI) or to other programs (e.g., via a programmatic interface).” MS-1005, [0078]. Thus, in implementations of the Hirschfeld-Smith-Vasilevsky System, provisioning requests may be issued via a management API forming part of the virtualization system’s control-plane interface for provisioning and resource-allocation operations. MS-1005, [0074]-[0079]; MS-1003, ¶¶252-254. Further, as noted in [11.6], the Hirschfeld-Smith-Vasilevsky System would have enabled

provisioning of a selected logical server that functions as virtual storage and utilizes virtualized storage constructs mapped to underlying physical storage devices (*virtual storage*). MS-1005, [0011], [0150], [0166]; MS-1006, 10:32-56; MS-1003, ¶¶252-254.

(f) Claim 13

See [6]; MS-1003, ¶¶255-257.

(g) Claim 14

The Hirschfeld-Smith-Vasilevsky System would have enabled a user (via a management-plane interaction used for requesting provisioning) to *receiv[e]* a listing of available virtual storage resources (*inventory of the virtual storage*) presented for selection/assignment in connection with provisioning operations. MS-1006, 6:28-63, 7:32; MS-1005, [0077]-[0079], [0082]-[0084], [0150]; MS-1003, ¶¶258-261.

As noted in [13], Hirschfeld provides the user-facing management pathway (e.g., through the SCM's user interface) through which a user submits requests and receives system-provided information associated with accessing and managing logical servers (*virtual assets*). MS-1006, 6:28-63, 7:32-48. Vasilevsky, in turn, describes that unassigned resources may be “grouped into a ‘pool’ of unassigned resources and presented to an administrator or program as being available for assignment.” MS-1005, [0082]. In the context of Vasilevsky's virtualization

management, this pool/presentation of unassigned resources includes storage-related resources because Vasilevsky expressly includes “storage” among the resources allocated/deallocated to virtual servers and describes a “mapping database that defines mapping of virtual resources to physical hardware,” including physical storage devices such as “disks.” MS-1005, [0072], [0150], [0166]. Thus, in the Hirschfeld-Smith-Vasilevsky System, when the user requests provisioning of virtual storage, the system would present and the user would receive an inventory of available virtual storage resources (e.g., available virtual storage entities/logical storage units available for assignment) to support the requested provisioning. MS-1003, ¶¶258-261.

H. GROUND 2D

1. Hirschfeld-Smith-Fujino

The Hirschfeld-Smith-Fujino System would have predictably combined Hirschfeld’s virtualization environment with Fujino’s well-known Storage Area Network (SAN) configuration and management techniques for logically representing network connectivity. MS-1003, ¶¶126-130; §III.C.1(d). A POSITA would have been motivated to implement this combination given Hirschfeld’s recognition that the shared network “may be segmented into multiple network fabrics” and that, when logical servers are moved, “network configuration changes may be required to maintain connectivity,” including that “network hardware

mappings (e.g., port mappings) are modified as necessary to maintain the logical net configuration.” MS-1006, 8:39-64, 15:10-17; MS-1003, ¶¶126-130. This would have led a POSITA to look to well-known techniques for maintaining logical network connectivity during reconfiguration events (including representing and managing paths/ports/switches in software), which are addressed in Fujino. MS-1003, ¶¶126-130.

As previously discussed, Fujino confirms that, by the Critical Date, it was well-known to manage SAN connectivity using maintained SAN configuration information and related management information identifying physical network devices (e.g., Fibre Channel switches and endpoint ports) along communication paths between host devices and storage control devices. MS-1008, 5:31-45, 6:27-36, 6:40-49, 7:35-43; MS-1003, ¶¶126-130. A POSITA implementing Hirschfeld’s SCM-based control of logical servers (including maintaining logical network configuration through port mappings) would have been motivated to incorporate Fujino’s SAN configuration teachings to improve predictability and controllability of network connectivity for storage access and related communications in the virtualized environment. MS-1006, 15:10-17; MS-1008, 7:24-34, 15:21-35, 27:27-41; MS-1003, ¶¶126-130.

Further, Fujino’s SAN configuration teachings (e.g., software-defined representations of communication paths and port-based connectivity to storage

resources) complement Hirschfeld's SCM-controlled virtualization paradigm, without requiring changes to the fundamental concept of running isolated logical servers on shared physical infrastructure. MS-1003, ¶¶126-130. In this regard, a POSITA would have reasonably expected success because the combination involves applying well-known SAN management mechanisms (e.g., maintained configuration information describing port topology and communication paths) to a similar type of virtualized, networked environment already contemplated within Hirschfeld for maintaining logical network connectivity during reconfiguration (i.e., modifying port mappings and related network configuration to preserve the logical net configuration). MS-1006, 15:10-17; MS-1008, 6:27-36, 7:35-43, 27:27-41; MS-1003, ¶¶126-130.

2. Analysis

(a) Claim 9

The Hirschfeld-Smith-Fujino System would have provided a provisioned logical server (*virtual asset*) configured to utilize a software-defined representation of network connectivity (e.g., SAN configuration information) (*logical network resource*) that specifies connectivity between host computers and storage systems by identifying physical devices along the communication path (e.g., fiber channel switches and endpoint ports). MS-1006, 6:28-63, 10:32-56; MS-1008, 6:27-36, 7:35-43, 17:40-54; MS-1003, ¶¶262-265.

As noted in [1.1]-[1.5], the Hirschfeld-Smith System describes providing user access to logical servers (*virtual assets*) deployed on physical servers under control of the SCM (*control computer*), with resources (including *network resource[s]*) virtualized to maintain virtual independence among logical servers. MS-1006, 6:28-63, 10:32-56; MS-1003, ¶262. In the Hirschfeld-Smith-Fujino System, the combination would have further incorporated Fujino's Storage Area Network (SAN) architecture to provide managed network connectivity (including storage-network connectivity) between such physical servers (hosting logical servers) and underlying storage devices. MS-1008, 5:31-45; *see* Ground 1E (addressing [9]). A POSITA would have understood and/or found it obvious that SAN configuration information is a logical (software-defined) abstraction that identifies underlying physical network devices (ports, switches) to define the communication path used by the system, and thus, represents an example of a *logical network resource* provided as part of providing virtual assets in the Hirschfeld-Smith environment. MS-1008, 6:27-36, 7:35-43; MS-1003, ¶¶263-65.

(b) Claim 10

The Hirschfeld-Smith-Fujino System would have provided a software-defined network path representation (e.g., maintained in the SAN configuration information and related communication path information) (*logical network unit*) that is associated with (*mapp[ed]* to) the underlying network hardware (*physical*

network device), such as the physical ports of the storage control device and the intervening fiber channel switch hardware used to implement the communication path. MS-1006, 15:10-17; MS-1008, 6:27-36, 6:40-49, 7:35-43, 27:27-41; MS-1003, ¶266.

As noted in [9], Fujino describes maintaining SAN configuration information that defines connectivity between endpoints by identifying physical devices along the path. MS-1008, 6:27-36, 7:35-43; *see* Ground 1E (addressing [9]). A POSITA would have understood and/or found it obvious that the software-defined representation of the path (*logical network unit*) represents a logical abstraction that identifies (and thereby maps to) the corresponding underlying physical network devices (ports and FC-SWs) that implement the communication path. MS-1008, 6:27-36, 6:40-49, 7:35-43, 27:27-41; MS-1003, ¶¶267-69.

IV. CONCLUSION

The Challenged Claims are unpatentable. Petitioner authorizes charge of fees to Deposit Account No. 06-1050.

V. MANDATORY NOTICES

A. Real Party-In-Interest Under 37 C.F.R. § 42.8(b)(1)

Petitioner (Microsoft Corporation) is the real party-in-interest.

B. Related Matters Under 37 C.F.R. § 42.8(b)(2)

The '915 patent is the subject of civil action *ToutVirtual, Inc. v. Microsoft Corporation*, 2-25-cv-00943 (EDTX) filed September 8, 2025. The '915 patent is subject to a terminal disclaimer with respect to U.S. Pat. No. 9,444,762. MS-1002, 235, 310-11. Microsoft is not aware of any disclaimers, reexamination certificates or petitions for *inter partes* review for the '915 patent. Microsoft is not aware of any civil actions concerning the '915 patent.

C. Lead And Back-Up Counsel Under 37 C.F.R. § 42.8(b)(3)

Petitioner provides the following designation of counsel.

Lead Counsel	Backup Counsel
W. Karl Renner, No. 41,265 Fish & Richardson P.C. 60 South Sixth Street Suite 3200 Minneapolis, MN 55402 Tel: 202-783-5070 Fax: 877-769-7945 Email: IPR42342-0156IP1@fr.com	Usman Khan, Reg. No. 70,439 Ryan Chowdhury, Reg. No. 74,466 Fish & Richardson P.C. 60 South Sixth Street Suite 3200 Minneapolis, MN 55402 Tel: 202-783-5070 Fax: 877-769-7945 IPR42342-0156IP1@fr.com

D. Service Information

Please address all correspondence and service to the address listed above.

Petitioner consents to electronic service by email at IPR42342-0156IP1@fr.com.

Respectfully submitted,

Dated: March 6, 2026

/W. Karl Renner/

W. Karl Renner, Reg. No. 41,265
Usman Khan, Reg. No. 70,439
Ryan Chowdhury, Reg. No. 74,466
Fish & Richardson P.C.
60 South Sixth Street, Suite 3200
Minneapolis, MN 55402
T: 202-783-5070
F: 877-769-7945

(Control No. IPR2026-00295)

Counsel for Petitioner

CERTIFICATION UNDER 37 CFR § 42.24

Under the provisions of 37 CFR § 42.24(d), the undersigned hereby certifies that the word count for the foregoing Petition for Inter Partes Review totals 13,926 words, which is less than the 14,000 allowed under 37 CFR § 42.24.

Dated: March 6, 2026

/W. Karl Renner/

W. Karl Renner, Reg. No. 41,265

Usman Khan, Reg. No. 70,439

Ryan Chowdhury, Reg. No. 74,466

Fish & Richardson P.C.

60 South Sixth Street, Suite 3200

Minneapolis, MN 55402

Tel: 202-783-5070

Fax: 877-769-7945

Counsel for Petitioner

CERTIFICATE OF SERVICE

Pursuant to 37 CFR §§ 42.6(e)(4)(i) et seq. and 42.105(b), the undersigned certifies that on March 6, 2026, a complete and entire copy of this Petition for Inter Partes Review and all supporting exhibits were provided by Federal Express, to the Patent Owner, by serving the correspondence address of record as follows:

DLA PIPER LLP (US)
3203 Hanover Street
Suite 100
PALO ALTO, CA 94304
UNITED STATES
(650) 833-2000

/Crena Pacheco/
Crena Pacheco
Fish & Richardson P.C.
60 South Sixth Street, Suite 3200
Minneapolis, MN 55402
pacheco@fr.com