IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent of:	Vipul Pabari	
U.S. Patent No.:	10,810,050	Attorney Docket No.: 42342-0155IP1
Issue Date:	October 20, 2020	
Appl. Serial No.:	15/201,087	
Filing Date:	July 1, 2016	
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. 10,810,050 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 Interpretation 2
1. "control center computer" 3
2. "automatically provisioning"/"automatically provision" 3
3. "without regard to…" 3
4. "virtual assets" 4
5. "physical resources" 4
6. "apparatus" 4
7. "network resources" 4
8. "software instantiations of computer systems" 4
9. "real time performance metrics" 5
10. "provisioning"/"provisioned" 5
11. "managed"/"manages" 5
12. "matches" 6
D. Level of Ordinary Skill in the Art 6

II. THE '050 PATENT 6
A. Specification 6
B. Relevant Prosecution 7

III. GROUNDS 8
A. GROUND 1A 8
1. Khandekar 8
2. Analysis 10
B. GROUND 1B 23
1. Khandekar Under §103 23
2. Analysis 25
3. Claims 11, 12, 14, 16 38
C. GROUND 1C 40
1. Khandekar-Vasilevsky 40
2. Analysis 47
D. GROUND 1D 49
1. Khandekar-Tremain 49
2. Analysis 51
E. GROUND 2A 52
1. Hirschfeld 52

2. Analysis 53
F. GROUND 2B 66
1. Hirschfeld-Vasilevsky 66
2. Analysis 69
G. GROUND 2C 69
1. Hirschfeld-Tremain 69
2. Analysis 71
H. GROUND 3 73
1. Lack of Priority Date Entitlement 73
2. Pabari-Bonazzola 76
3. Analysis 79
IV. CONCLUSION 82
V. MANDATORY NOTICES 82
A. Real Party-In-Interest Under 37 C.F.R. § 42.8(b)(1) 82
B. Related Matters Under 37 C.F.R. § 42.8(b)(2) 82
C. Lead And Back-Up Counsel Under 37 C.F.R. § 42.8(b)(3) 82
D. Service Information 83

LIST OF EXHIBITS

MS-1001	U.S. Patent No. 10,810,050 to Pabari ("the '050 Patent")
MS-1002	Prosecution History of the '050 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	U.S. Pat. App. Pub. No. 2014/0344462 ("Pabari")
MS-1013	U.S. Pat. App. Pub. No. 2015/0347167 ("Bonazzola")
MS-1014	U.S. Provisional App. No. 60/708,473 ("Pabari-Provisional")
MS-1015	U.S. Pat. App. No. 15/201,087 ("Pabari-Utility-Application")
MS-1016-1049	RESERVED

MS-1050	Complaint, *ToutVirtual, Inc. v. Microsoft Corporation*, 2-25-cv-00943, EDTX (filed Sep. 8, 2025)
MS-1051	Docket Control Order, *ToutVirtual, Inc. v. Microsoft Corporation*, 2-25-cv-00943, EDTX (issued Feb. 19, 2026)
MS-1052	Appendix C, INFRINGEMENT CHART U.S. Patent No. 10,810,050, *ToutVirtual, Inc. v. Microsoft Corporation*, 2-25-cv-00943, EDTX

LISTING OF CHALLENGED CLAIMS

Claim 1	
[1.pre]	A method for on demand provisioning of virtual asset via a self-service portal in a virtualization system, the method comprising:
[1.1]	providing a virtualization system that has an inventory that includes physical resources and virtual assets managed by a virtualization layer running on the physical resources;
[1.2]	wherein virtual assets are software instantiations of computer systems;
[1.3]	receiving a virtual asset provisioning request that specifies parameters to be considered for the virtual asset provisioning request;
[1.4]	receiving data on inventory available in the virtualization system;
[1.5]	automatically provisioning of a particular virtual asset to a particular physical resource in the virtualization system that matches the parameters in the virtual asset provisioning request;
[1.6]	assigning the provisioned virtual asset to the virtual asset provisioning request; and
[1.7]	providing user access to the provisioned virtual asset.
Claim 2	
[2]	The method of claim 1 further comprising determining one or more performance metrics of the provisioned virtual asset.
Claim 3	
[3]	The method of claim 1 further comprising providing a user interface for submitting the virtual asset provisioning request.

Claim 4	
[4]	The method of claim 3, wherein providing the user interface further comprises providing a user interface to access one or more images in a virtual asset image repository.
Claim 5	
[5]	The method of claim 2, wherein determining the one or more performance metrics further comprises determining one or more real time performance metrics.
Claim 6	
[6]	The method of claim 1, wherein the inventory further comprises one or more application programs.
Claim 7	
[7]	The method of claim 6, wherein one of the application programs comprises one of a network load balancer, a router, a firewall and a storage server.
Claim 8	
[8]	The method of claim 6, wherein one of the application programs comprises a network server.
Claim 9	
[9.pre]	An apparatus that performs automatic management of a virtualization environment in a computer network comprising:
[9.1]	a virtualization environment having a computer device with a processor and memory and a control center computer coupled to the computer device,
[9.2]	the virtualization environment having an inventory that includes network resources and virtual assets being software instantiations of computer systems hosted on the computer device and a virtualization layer executed by the computer device that manages the virtual assets;

[9.3]	the control center computer executing a virtual mapping layer that determines an inventory of network resources and virtual assets available to the virtualization environment,
[9.4]	wherein the virtual mapping layer determines the inventory without regard to processors, operating systems, virtualization platforms, and application software of the virtualization environment; and
[9.5]	the control center computer executing a provisioning manager that communicates with the virtual mapping layer,
[9.6]	the processor being configured to receive a virtual asset provisioning request that specifies parameters to be considered for the virtual asset provisioning request,
[9.7]	automatically provision a particular virtual asset to a particular physical resource in the virtualization system that matches the parameters in the virtual asset provisioning request,
[9.8]	assign the provisioned virtual asset to the virtual asset provisioning request and provide user access to the provisioned virtual asset.
Claim 10	
[10]	The apparatus of claim 9, wherein the virtualization environment determines one or more performance metrics of the provisioned virtual asset.
Claim 11	
[11]	The apparatus of claim 9, wherein the provisioning manager provides a user interface for submitting the virtual asset provisioning request.

Claim 12	
[12]	The apparatus of claim 11, wherein the provisioning manager provides a user interface to access one or more images in a virtual asset image repository.
Claim 13	
[13]	The apparatus of claim 10, wherein the virtualization environment determines one or more real time performance metrics.
Claim 14	
[14]	The apparatus of claim 9, wherein the inventory further comprises one or more application programs.
Claim 15	
[15]	The apparatus of claim 14, wherein one of the application programs comprises one of a network load balancer, a router, a firewall and a storage server.
Claim 16	
[16]	The apparatus of claim 14, wherein one of the application programs comprises a network server.

Microsoft Corporation ("**Petitioner**" or "**Microsoft**") petitions for *Inter Partes* Review ("**IPR**") of claims 1-16 ("**the Challenged Claims**") of U.S. Patent No. 10,810,050 ("**the '050 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 '050 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

This petition sets forth challenges on grounds noted below, and offers the supporting testimony of Dr. Michael Mitzenmacher. MS-1003, ¶¶19-20.

Ground	Claims	Basis for Unpatentability
1A	1, 3-4, 6-8	§102: Khandekar
1B	9, 11, 12, 14, 16	§103: Khandekar
1C	2, 5, 10, 13	§103: Khandekar, Vasilevsky
1D	7, 15	§103: Khandekar, Tremain

Ground	Claims	Basis for Unpatentability
2A	1, 3	§103: Hirschfeld
2B	2, 5	§103: Hirschfeld, Vasilevsky
2C	6, 7	§103: Hirschfeld, Tremain
3[1]	9-16	§103: Pabari, Bonazzola

The ’050 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. MS-1001, Cover. As noted below in §III.H, these priority applications fail to disclose features of at least claims 9-16, which were added by amendment, foreclosing proclaimed eligibility for these claims and leaving them to the 10/2/2017 application filing date. Regardless, the prior art references Khandekar, Vasilevsky, Tremain, and Hirschfeld qualify as prior art under the 8/15/2005 date, and Pabari and Bonazzola qualify as prior art under the 10/2/2017 date.

C. Claim Interpretation

“[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.

[1] Claims 9-16 lack priority to the earliest possible priority date of 8/15/2005. *See* §III.H.

Cir. 2011). In this case, the asserted Grounds render the Challenged Claims obvious under any reasonable construction, including plain and ordinary meaning. For clarity, accounting for the context offered by the ’050 patent, Microsoft proposes the following constructions, which are consistent with disclosure found in the ’050 patent, and therefore, demonstrate subject matter that falls within the metes and bounds of these terms. The mappings included below in §III are not inconsistent with these constructions. MS-1003, ¶63.

1. **“control center computer”**

A POSITA would have understood that the term “control center 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.” *See* MS-1001, 5:62-6:31, 7:1-14; MS-1003, ¶63.

2. **“automatically provisioning”/“automatically provision”**

A POSITA would have understood that the claimed “automatically provisioning” and “automatically provision” at least encompasses “determining without human intervention.” *See* MS-1001, 4:16-39, 8:21-45; MS-1003, ¶63.

3. **“without regard to…”**

A POSITA would have understood that the claimed “without regard to…” at least encompasses “irrespective of any.” *See* MS-1001, 4:11-36, 6:42-63; MS-1003, ¶63.

4. **"virtual assets"**

A POSITA would have understood that the claimed "virtual assets" at least encompasses "software instantiations of equipment objects that exist only within a virtual environment." *See* MS-1001, 6:32-37; MS-1003, ¶63.

5. **"physical resources"**

A POSITA would have understood the claimed "physical resources" to encompass "physical devices on the network, such as computers, printers, routers, and other tangible hardware." *See* MS-1001, 6:32-37; MS-1003, ¶63.

6. **"apparatus"**

A POSITA would have understood that the claimed "apparatus" at least encompasses "a system with two or more devices." *See* MS-1001, 3:50-4:3, 9:45-67, FIGS. 1-3; MS-1003, ¶63.

7. **"network resources"**

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," as these are the network resources disclosed by the '050 patent. *See* MS-1001, 2:10-26; MS-1003, ¶63.

8. **"software instantiations of computer systems"**

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-1001, 3:9-19, 9:62-10:17; MS-1003, ¶63.

9. **"real time performance metrics"**

A POSITA would have understood that the claimed "real time performance metrics" at least encompasses "measurements regarding usage of virtual assets obtained on-demand." *See* MS-1001, 14:12-27; MS-1003, ¶63.

10. **"provisioning"/"provisioned"**

A POSITA would have understood that the claimed "provisioning" and "provisioned" at least encompass "associating a virtual asset to a particular physical computer resource." *See* MS-1001, 20:12-32; MS-1003, ¶63.

11. **"managed"/"manages"**

The '050 patent states that "management of computer network virtualization environments is provided by performing one or more functions," including:

> (1) identification and management of network resources and virtual assets, (2) provisioning of virtual assets in response to network workflow demands, (3) dynamic deployment of virtual assets across the computer network, (4) performance measurement and reporting of resources and virtual assets, and (5) planning and forecasting of resource demands and asset utilization of the virtualization environment, such that the functions are carried out without regard to processors, operating systems, virtualization platforms, and

application software of the virtualization environment.

MS-1001, 4:16-39. A POSITA would have understood that the claimed "managed" and "manages" at least encompasses the management features noted in the passage above. MS-1003, ¶63.

12. **"matches"**

A POSITA would have understood that the claimed "matches" at least encompasses "equating to an exact match, not meeting a minimum threshold." *See* MS-1001, 18:22-37; MS-1003, ¶63.

D. Level of Ordinary Skill in the Art

A person of ordinary skill in the art ("POSITA") relating to the subject matter of the '050 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, ¶¶21-23. Additional education could substitute for professional experience, and professional experience could substitute for education. *Id*.

II. THE '050 PATENT

A. Specification

The '050 patent provides "automated management of network-based virtual environments using Control Centers that "manage[ ] multiple resources" (e.g.,

blade servers and other network-based equipment) and that "can be configured to support a variety of virtual assets, including desktop computer environments, data storage, application servers, routers, and the like." MS-1001, 6:38-59, 7:13-32, FIG. 1-3; MS-1003, ¶¶51-53.

B. Relevant Prosecution

The '050 patent issued from examination that featured three Office Actions advancing rejections based on Vasilevsky and U.S. Pat. App. No. 2003/0236882 ("Yan"). MS-1002, 97-102, 141-57, 193-207. Khandekar surfaced during examination, but the Examiner failed to apply Khandekar in any ground of rejection. *Id.* As explained in §§III.A.2(a) and III.B.2(a), *infra*, Khandekar discloses the "an inventory that includes physical resources and virtual assets managed by a virtualization layer running on the physical resources" (recited in [1.1], [9.2]) and "virtual assets [that] are software instantiations of computer systems") (recited in that [1.2], [9.2])—which are the features that Applicant repeatedly urged were distinguishable over the prior art of record (*see, e.g.*, MS-1002, 133-34, 189-90, 237-38), and which led to allowance. *Id.*, 268; MS-1003, ¶¶54-62.

III. GROUNDS

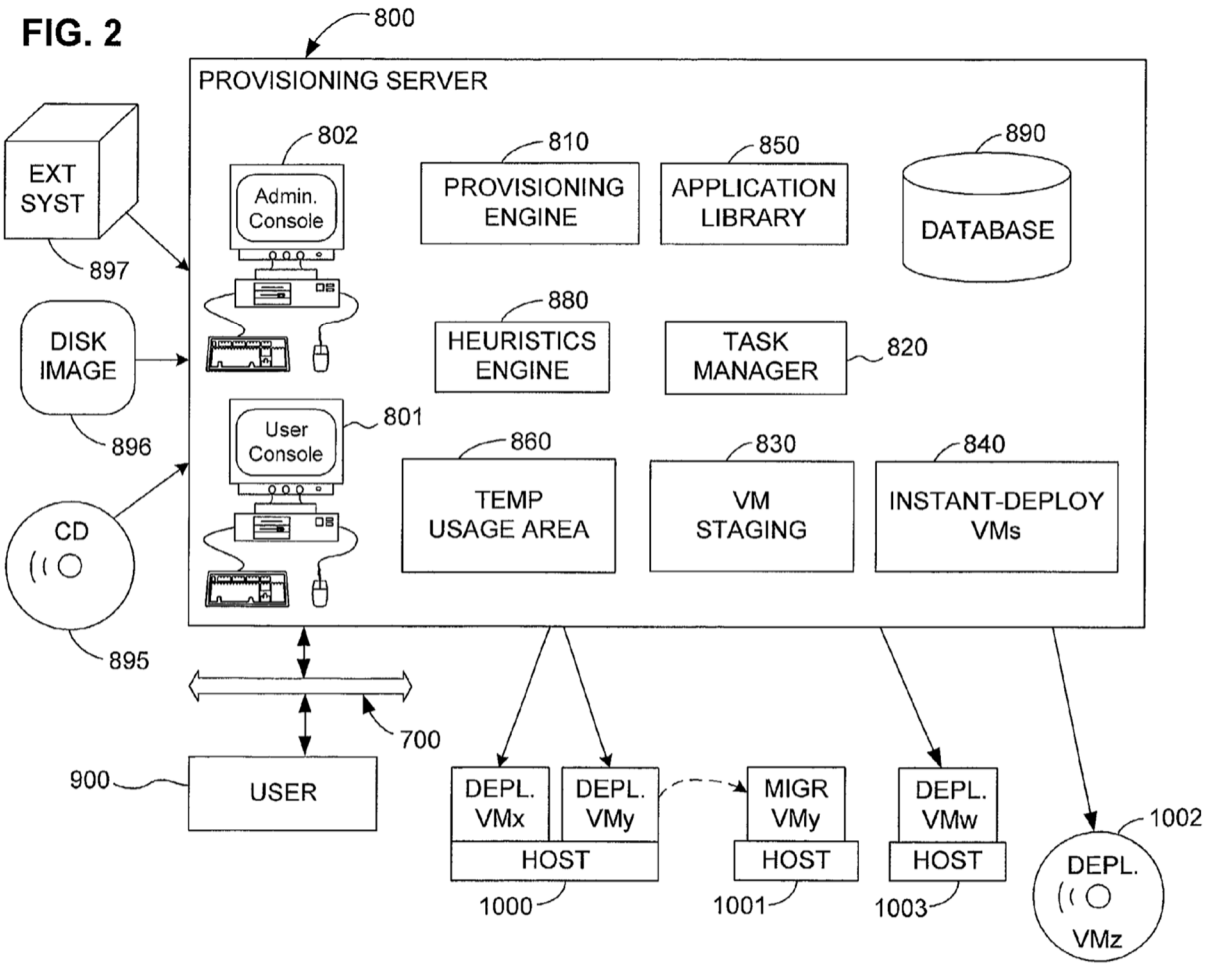
A. GROUND 1A

1. Khandekar[2]

Khandekar discloses 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, 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 Khandekar's provisioning system:

[2] General descriptions provided for this and other references and combinations thereof are incorporated into each subsection and mapping of the claims that include citations to these references.



MS-1004, FIG. 2.

As shown above, a centralized VM provisioning environment allows a user 900 to communicate with the provisioning server. MS-1004, 9:46-64. The provisioning server includes a front-end user console 801 through which the user can "submit the selected options." *Id.*, 10:15-26. The VM provisioning server includes an administration console "to monitor the operational state" and "maintain the database." *Id.*, 9:46-64, 10:15-26. The provisioning server processes user-defined configuration characteristics using a rules/heuristics component (e.g.,

heuristics engine), which "select[s] the most appropriate VM from a library 830 of staged VMs" and "selects 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), with "any number of VMs … deployed and loaded on a single host." MS-1004, 10:50-60, 12:57-13:8; MS-1003, ¶¶79-84.

2. **Analysis**

(a) Claim 1

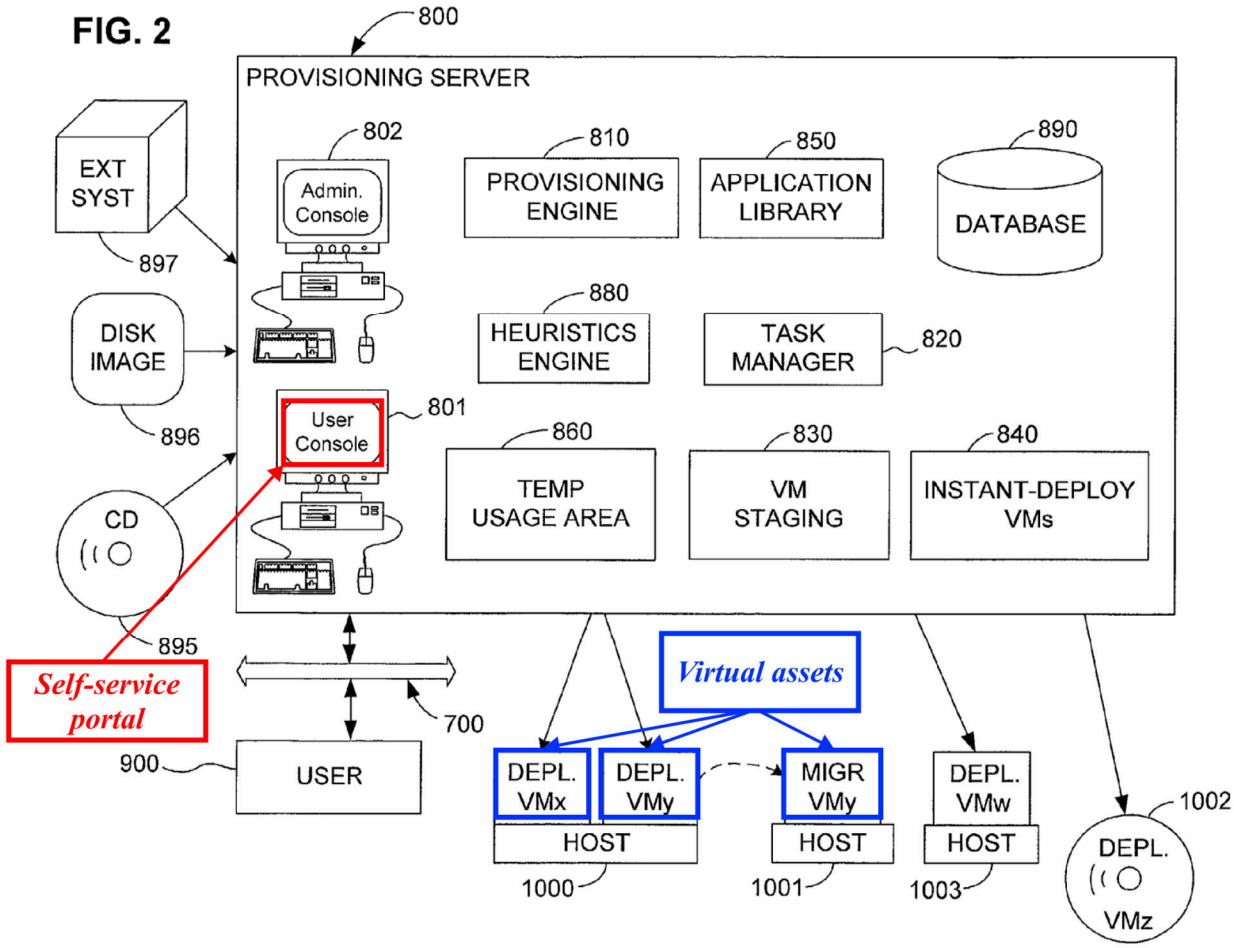
[1.pre]

To the extent the preamble is limiting, Khandekar's method satisfies [1.pre]. Khandekar's system implements a process for automatically configuring and deploying virtual machines (***method for on demand provisioning***) of a "Virtual Machine" (or "VM") that exists only in a virtualization environment (***virtual asset***). Khandekar's system receives user input via a user console 801 accessed by a web browser (***self-service portal***) within a provisioning environment (e.g., VM provisioning server 800 environment interacting with hosts 1000, 1001) (***virtualization system***). MS-1004, 1:12-14, 1:64-2:14, 5:25-28, 9:20-25, 9:46-64, 11:20-35, 13:66-14:6, 15:29-39, 21:26-31, 22:35-51; MS-1003, ¶143-145.

As discussed in §III.A.1, Khandekar is focused on provisioning quickly in response to user-specified configuration requirements, using an instantaneous

automated workflow to achieve ***on demand provisioning*** that "with no need for human intervention." *Id.*, 5:25-28. Such automated workflows (e.g., "instant-deploy" and automated "custom-built" workflows) are examples of ***on demand*** operations because they are initiated immediately upon user request and executed automatically by the provisioning engine. *Id.*, 9:20-25, 15:29-39, 21:26-31; MS-1003, ¶144.

Khandekar's user console is "preferably a web server, with which the user interacts with[in] any conventional web browser." MS-1004, 13:66-14:3. Through this interface, a user initiates self-service by "log[ging] in to the provisioning server" to configure/customize a VM, thereby identifying compute needs and initiating the provisioning process by the networked provisioning server and host environment. MS-1004, *Id.*, 9:46-64, 14:4-6, 22:35-51; MS-1003, ¶145, FIG. 2 (annotated below).



MS-1004, FIG. 2.[3]

[1.1]

Khandekar's system (***virtualization system***) includes "hosts" that are

[3] Annotations to figures throughout this petition are shown in colored markings and text. Also, references throughout this petition to specific reference numerals are intended as merely exemplary, and not to suggest limiting the discussion or analysis to the aspect corresponding to the identifier that is referenced.

physical devices on a network (***physical resources***) and VMs (e.g., "Instantly deployable VMs" or "Staging VMs") (***virtual assets***). A POSITA would have understood "resources and assets in the virtualization system and information thereof," and thus, information stored in database 890 is an example of ***inventory***. MS-1003, ¶147. These elements are managed by a "virtual machine monitor" (or "VMM") (***virtualization layer***) running on the hosts (***running on the physical resources***). MS-1004, 7:7-10, 7:28-38, 8:20-35, 9:26-32, 10:27-42, 13:17-29, 18:2-35; MS-1003, ¶¶146-150.

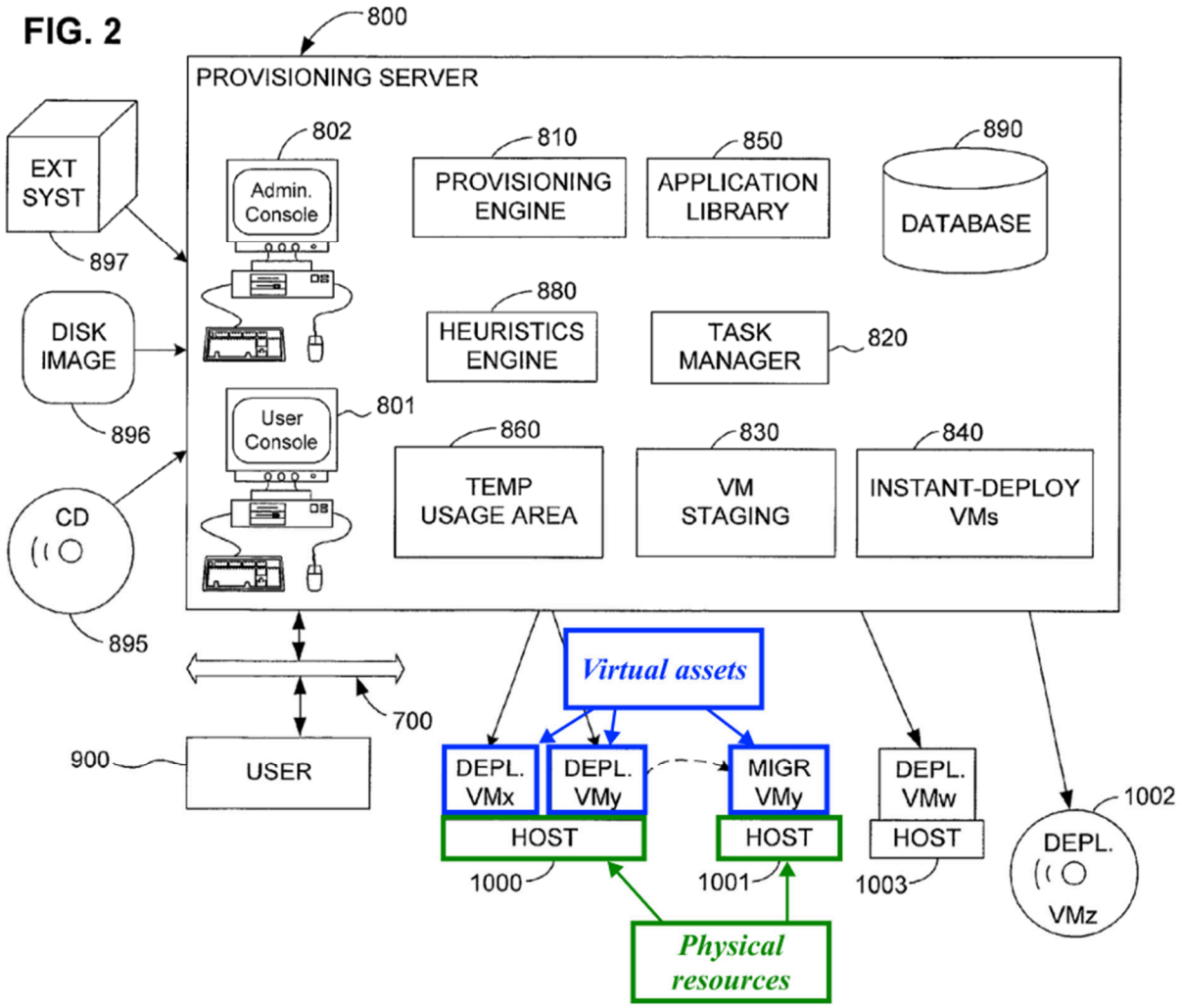
As noted in [1.pre], Khandekar includes a provisioning server that manages a plurality of "Host" computers. MS-1004, 9:26-32, 10:27-42. The system maintains a database 890, which serves as a "central repository of data required by the various components." *Id.*, 18:2-5. The repository includes data structures for both "hosts" (***physical resources***) and "Staging VMs" or "Instantly deployable VMs" (***virtual assets***). *Id.*, 18:6-35. The database also keeps "information about the hosts available" and indexes the "VM configuration" for virtual assets. *Id*.

A virtual machine monitor (VMM) is "installed on each of a plurality of hardware hosts." *Id.*, 13:17-29. The VMM manages VMs by identifying and managing network resources and virtual assets, e.g., by "schedul[ing] and handl[ing] all requests by its VM for machine resources." *Id.*, 7:28-38. The VMM is "usually a thin piece of software that runs directly on top of the host." *Id*. Thus,

in Khandekar, the hosts (***physical resources***) run a VMM (***virtualization layer***) to support the virtual machines (***virtual assets***) tracked in the database (***inventory***). MS-1003, ¶¶146-150.

[1.2]

As noted in [1.1], Khandekar describes a virtual machine (VM) as "a software abstraction—a 'virtualization'—of an actual physical computer system." MS-1004, 6:24-27. Khandekar VMs (***virtual assets***) 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, by providing software that emulates the characteristics and functions of computer systems, e.g., physical devices that connect to computers, Khandekar's VMs include software that are examples of ***software instantiations of computer systems***. MS-1003, ¶¶151-153.



MS-1004, FIG. 2

[1.3]

As noted in [1.pre], in Khandekar, a user interacts with the user console 801 to initiate provisioning. MS-1004, 9:46-64. More specifically, "[w]hen the user submits a request to provision a new VM, the provisioning engine requests that a new task be created" (***receiving a virtual asset provisioning request***). *Id*., 20:64-67. This request includes specific configuration details explicitly identified, as

parameters. For example, for custom-built VMs, Khandekar contemplates the "user specif[ying] various (virtual) hardware components," with "[t]he various parameters that the user will normally need to specify" including "[t]he main memory (RAM), "number of disks," and "size of each disk." *Id.*, 14:31-64. Additionally, the user specifies the "desired OS configuration, for example, hostname, IP address." *Id.*, 9:65-10:14. A POSITA would have understood that these user-selected hardware and software characteristics constitute "***parameters to be considered***" because they are furnished to the provisioning server by the user to inspire their subsequent use as configuration input, to deploy the virtual machine. MS-1003, ¶¶154-155.

[1.4]

As noted in [1.1], Khandekar's database 890 maintains an inventory including, for example, "[i]nformation about the current state of the different hosts [to be] preferably input dynamically into a database or other data structure in the heuristics engine 880." MS-1004, 13:1-4. Database 890 also includes "information about the hosts available to deploy VMs on," such as "[h]ardware configuration of the host (memory size, disk space, etc.)." *Id.*, 18:33-39.

To receive certain inventory data, the heuristics engine 880 is configured to "[s]can the list of available host machines 1000, 1001, etc.," and "determine the one with enough memory and disk space" by "querying the host via an API

provided to manage VMs." *Id.*, 20:54-58. The system receives the inventory data in response when the host operates to "[r]eturn the configuration of the host" back to the provisioning server. *Id.*, 20:61-62. By structurally querying the physical hosts via an API and by having physical hosts return their current configuration information, Khandekar teaches receiving ***data on inventory available in the virtualization system***. MS-1003, ¶¶156-158.

[1.5]

Khandekar automatically configures and deploys a specific VM instance (e.g., VMx) (***particular virtual asset***) to a selected host (e.g., Host 1000) (***particular physical resource***) in the provisioning environment (***virtualization system***) without human intervention (***automatically provisioning***). A user may specify certain requirements within the provisioning request (e.g., operating system, desired RAM) (***parameters in the virtual asset provisioning request***), and the system provisions the VM to match the specified parameters (***match[ed]***) (e.g., VM having a specific OS with the same amount of RAM as the desired RAM). MS-1004, 14:26-49; MS-1003, ¶159-160.

As noted in [1.3], when providing information for a provisioning request, the user specifies information for "various (virtual) hardware components of a VM in order for the provisioning server to be able to create the VM." MS-1004, 14:31-36. Examples of such inputted information include "OS selection" (*id.*, 14:26-29)

and a desired amount of "main memory (RAM)" (*id.*, 14:31-49). To execute provisioning, Khandekar teaches targeting a VM with the same hardware configuration as that specified in the provisioning request, e.g., the same OS and the same "desired amount of memory." *Id.* In this way, Khandekar teaches ***match[ing]***, for example, the OS and RAM amount of the VM to the types specified in the provisioning request (e.g., selected OS, desired amount of RAM). MS-1003, ¶160.

[1.6]

As noted in [1.3], the provisioning process in Khandekar is initiated by a user request. Upon receiving this request, "a new transaction is started in the task manager 820 and the user is returned a task ID." MS-1004, 15:29-39; *see* 10:61-67. This Task ID uniquely "identifies the task" associated with the specific provisioning request. *Id.*, 20:64-21:8. To process this request, the provisioning server "creates a temporary usage area 860 for the provisioning request that holds the user input as well as the various temporary data needed while the provisioning task is in progress." *Id.*, 10:61-67.

Specifically, "[f]or each VM in the process of being provisioned, a corresponding storage space 866 is preferably provided" within this temporary usage area. *Id.*, 19:20-25. The provisioning server maps the requested VM to this task by performing steps, such as "[c]reat[ing] a new temporary usage area 860 for

this task," storing the user's configuration choices, and creating the necessary files to "[c]opy the VM's config file and the disk files to the host." *Id.*, 21:39-42, 21:50. Thus, at least by linking the creation and deployment of the specific VM to the dedicated temporary usage area and task ID generated for the user's submission, Khandekar teaches ***assigning*** the provisioned VM (***provisioned virtual asset***) to the user-submitted request (***virtual asset provisioning request***). MS-1003, ¶¶161-163.

[1.7]

Khandekar teaches enabling a user to access provisioned VMs (***provisioned virtual assets***) deployed on and executed by the hosts via a user console, thereby ***providing user access*** to them. MS-1004, 9:65-10:14, 13:66-14:18, 15:29-39, 20:64-21:8, 21:9-15. For example, a user may use "remote access software" or "remote console software" to connect to a provisioned VM. MS-1004, 9:65-10:14, 13:66-14:18, 15:29-39, 20:64-21:8, 21:9-15; MS-1003, ¶¶164-166.

The user connects to the provisioning server through the user console, which is "preferably a web server…" MS-1004, 13:66-14:3. Through this connection, the user can, for example, 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.*; *see id.*, 9:65-10:14, 14:4-

18. 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. And, once the VM is deployed, "[t]he user can then access the newly created VM via any known remote access software," e.g., "…software available from VMware." *Id*., 14:4-18. Khandekar thereby provides a user associated with the request with the necessary connectivity to utilize the provisioned asset, which reflects an example of ***providing user access***. MS-1003, ¶¶165-166.

(b) Claims 3, 4, 6, 8

[3]

Khandekar provides a user console 801 that presents "dialogs" or web pages (***user interface***) to the user for "submit[ting] the selected options" and provisioning requests to a provisioning server (***submitting the virtual asset provisioning request***). MS-1004, 10:15-26, 13:66-14:3, 14:26-29, 15:16-28; MS-1003, ¶¶167-168; *supra* [1.3].

[4]

The user console (***user interface***) "is preferably a web server, with which the user interacts with any conventional web browser." MS-1004, 13:66-14:3.

Through this interface, "a list of available staged VMs" can be presented for display" and "the user may then select one of the VMs from the list." *Id.*, 10:15-26. These VMs are stored in designated repositories, such as a "library 830 of staged VMs" or an "instantly-deployable VMs repository 840." *Id.*, 10:27-42, 18:14-15. Deployable configurations are also associated with images. For example, instant-deploy VMs stored in repository 840 include a "suspend-to-disk image file 844." *Id.*, 17:31-49. Users may create a model VM from a "library of disk drive images" or install an OS from an "ISO image." *Id.*, 15:55-67. At least by presenting the user with a selectable list of these pre-built VMs and their underlying image files (***one or more images***) stored in a library/repository (e.g., repository 840, "library of disk drive images") (***virtual asset image repository***), Khandekar's user console represents an example of a ***user interface*** provided to ***access one or more images in a virtual asset image repository***. MS-1003, ¶¶169-170.

[6]

As noted in [1.1], Khandekar teaches maintaining an ***inventory*** using database 890. This database includes information, such as "Applications," which are a "data structure [that] keeps information about the applications available for installation in a custom-built VM." MS-1004, 18:24-26. This inventory data structure indexes an "Applications Library 850," which is "a repository of the

software and data needed to install an application 856 in a VM." *Id.*, 18:57-19:2. Through the user console 801, the system provides an "Application Selection Dialog" where the user "selects the applications he wants installed in the VM." *Id.*, 15:23-28. At least by including data structures tracking available software applications within the database, the database (***inventory***) thereby includes examples of ***one or more application programs***. MS-1003, ¶¶171-172.

[7]

Khandekar discloses an "Applications Library" (***application programs***) with code for "database servers" (***storage server***). MS-1004, 11:50-58, 23:11-20. Khandekar describes that a VM may run an application representing an "database server," which is a type of a ***storage server***. MS-1004, 11:50-58; *see id.*, 23:11-20. Moreover, because of the phrase "one of," claim 5 does not mandate providing end user access to each of the enumerated elements recited in the claim. MS-1003, ¶173.

[8]

As noted in [6], the system inventory includes an "Applications Library 850" containing various software applications (***application programs***). This library "contain[s] code for any collection of the myriad of existing applications, from web servers, database servers, spreadsheets and word-processing programs, to games and browsers." MS-1004, 11:50-58. Additionally, a network of

computers may be provisioned where "a developer wants to test his web application, which uses a web server on one computer, an application server on another, and a database server on yet another." *Id.*, 23:11-20. A POSITA would have understood that "web servers" and "database servers" are applications designed to process requests and deliver data across a network (e.g., Internet), and thus, represent examples of a ***network server*** under its plain and ordinary meaning. MS-1003, ¶¶174-175.

B. GROUND 1B

1. Khandekar Under §103

It would have been obvious for a POSITA to configure Khandekar's provisioning server to perform the functions of a centralized management apparatus, with a heuristics engine 880 and database 890 collectively supporting the orchestration of VM provisioning and thus functioning as a virtual mapping layer. MS-1003, ¶¶176-177 (citing MS-1004, 18:2-5, 20:32-42, 21:26-31). Indeed, the obviousness of this configuration would be readily apparent to a POSITA from Khandekar's description of functions performed by each component within its system, as those components and their functions are described discretely, consistent with a remote orientation. *Id.* (citing MS-1004, 9:26-45, 20:32-42, 21:26-34). For instance, the provisioning engine is described by Khandekar as "call[ing] the heuristics engine 880, to determine the VM's hardware configuration

and the host to deploy on" (*id.*, 21:33-34), which in turn matches user input "against the database 890" (*id.*, 20:32-42). And, the provisioning server relies on these/similar interactions between the provisioning engine 810, heuristics engine 880, and database 890 to determine where and how to allocate VMs to available host resources. *Id.*

As explained below, a POSITA would have further found it obvious that, when configured to operate as discussed above, the heuristics engine 880 and database 890 determine inventory, irrespective of any processors, operating systems, virtualization platforms, or application software. MS-1003, ¶¶178-180. In one implementation, where the VMM on each host "exports a known hardware interface to the VM deployed on it," Khandekar's VM deployments are "substantially independent of the actual physical hardware configuration." *Id.*, 13:17-35. Indeed, VMs may "migrate freely, even to other hardware and software platforms." MS-1004, 8:20-35.

Moreover, Khandekar's database 890 catalogs VMs across multiple operating systems (e.g., "Linux Red Hat, Windows 2000, Windows Me") and a broad range of applications (e.g., "web servers, database servers, spreadsheets and word-processing programs, to games and browsers"). MS-1004, 11:36-58. A POSITA would have understood that by cataloging VMs with diverse operating systems and application configurations as available options within the same

system, Khandekar's provisioning framework accommodates and supports provisioning across these different platforms. MS-1003, ¶178. This understanding is confirmed by Khandekar's disclosure instead resource availability parameters (e.g., "memory and disk space") are used for VM selection and not platform-specific characteristics. *Id*., 20:54-58. Thus, certain implementations of provisioning within Khandekar would have involved the provisioning server 800's management functions (e.g., provisioning) operating at an abstracted level, without regard to that is irrespective of any of the specific processors, operating systems, virtualization platforms, or application software on any given host. MS-1003, ¶¶179-180.

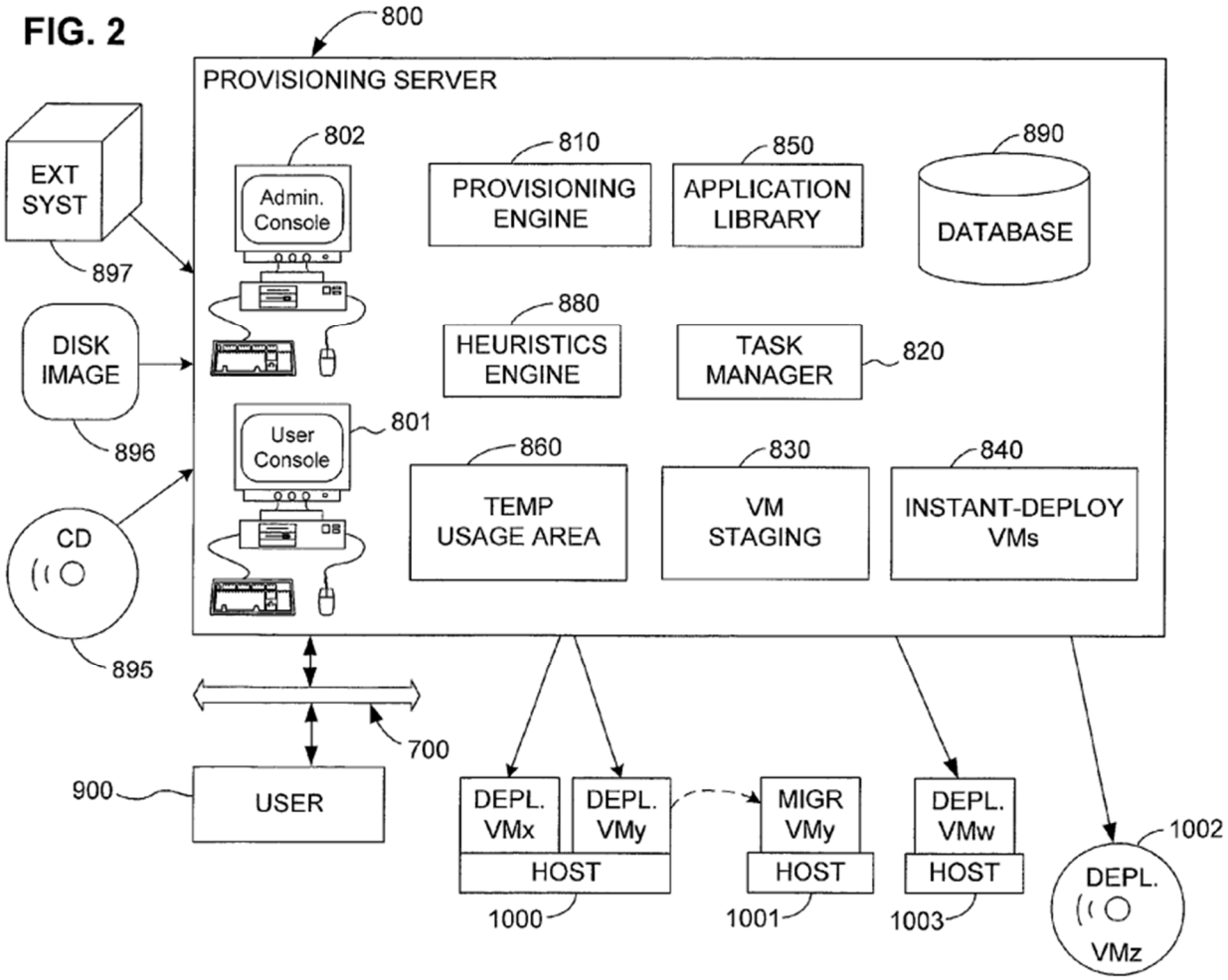
2. **Analysis**

(a) Claim 9

[9.pre]

As noted in [1.pre], Khandekar's system (***apparatus***) includes a "VM provisioning server 800" that is "configured as a network server" and implemented using "conventional network server technology." *Id.*, 9:26-45. The Khandekar's system includes two or more devices (e.g., server 800, host 1000). *Id.*, Fig. 2. Server automatically manages the environment by creating and deploying virtual machines "with no need for human intervention" (***perform[] automatic management***). *Id.*, 5:25-36. As shown in Khandekar's FIG. 2 below, provisioning

server is connected via a "network 700" (***virtualization environment***) to a plurality of "host" computers (e.g., 1000, 1001). *Id.*, 10:50-60. A POSITA would have understood from Khandekar that its implementations provide a similar networked arrangement of a management server and host computers hosting virtual machines, which exemplifies a ***virtualization environment***. MS-1003, ¶¶181-182.



MS-1004, FIG. 2.

[9.1]

Khandekar discloses a provisioning environment (***virtualization environment***) having one or more hosts containing a CPU and RAM (***computer device with a processor and memory***), and a provisioning server (***control center computer***) connected to the hosts via a network (***coupled to the computer device***). MS-1004, 5:59-6:12, 10:50-60; MS-1003, ¶¶183-189.

Khandekar describes deploying VMs onto "host" computers 1000, 1001. MS-1004, 10:50-60. The underlying hardware platform for these hosts includes "one or more central processors CPU(s) 110" and "system memory 112, typically implemented using any of the various RAM technologies." *Id.*, 5:59-67. Thus, in Khandekar, host computers would have each included a ***processor*** and ***memory***. MS-1003, ¶¶183-189.

A POSITA would have understood and found obvious that Khandekar's provisioning server 800 selects a VM for provisioning and coordinates deployments of the VMs for running on a host in a centralized manner for the system, consistent with the "Control Center computers" examples described in the '050 patent specification, and that recited by claim 9. MS-1003, ¶¶185-187 (citing MS-1004, MS-1001, 7:33-56, 8:17-40 and MS-1004, 10:27-60). Khandekar also describes the provisioning server as being "configured as a network server" and communicating with the host computers via a network (e.g., network 700). MS-

1004, 9:33-45. Together, provisioning server 800 and hosts 1000, 1001 operate as an integrated, cooperating system connected via network 700, where server 800's provisioning operations involve processors at the server and host each performing functions in support of coordinated activity between those devices. MS-1003, ¶188 (citing MS-1004, 21:26-64, 20:54-62).

This configuration is consistent with implementations of the '050 patent specification implicated by claim 9 (*see* MS-1001, 10:18-11:4, FIG. 3). For example, Figure 3 depicts a computer device 300 that includes a CPU 306 and memory 308, coupled to a virtualization layer 322 and a Control Agent 324. MS-1001, 9:39-61. The Control Agent 324 (and thus, its CPU) is described as facilitating "communication between the external Control Center components 204-214 (FIG. 2) and the Virtualization Layer 322 of the host machine" and providing "a common interface to the variety of virtualization software that might be available on the host machine." MS-1001, 10:18-42.

Like the '050 patent, Khandekar contemplates an architecture in which provisioning workflows are performed by a control element (e.g., provisioning server) in conjunction with a virtual asset (e.g., host 1000) in a cooperative distributed computer network arrangement. MS-1003, ¶¶188-189. Tellingly, in asserting infringement of claim 9, Patent Owner characterized a configuration in which a "processor, in conjunction with the control center computer," allegedly

performs the recited operations as being within claim 9's scope.[4] MS-1053, 100.

[9.2]

Khandekar provides a ***virtualization environment*** having an ***inventory*** in the form of database 890 that includes information identifying ***network resources*** and VMs (***virtual assets being software instantiations of computer systems***) hosted on the hosts (***computer devices***), with a VMM (***virtualization layer***) executed by the host that manages the VMs. MS-1004, 6:1-12, 7:7-27, 14:65-15:14, 18:2-41; MS-1003, ¶¶190-194.

As noted in [1.1], Khandekar's system maintains a database 890 that tracks available hosts and VMs. Further, the VMs are "software abstraction[s]" of physical computer systems (***software instantiations of computer systems***) that are deployed on and hosted by the hosts (***computer devices***). MS-1004, 6:24-42, 18:2-41. A VMM is installed on each host and "virtualizes all, or at least some of, the resources of the machine" (***virtualization layer executed by the computer device that manages the virtual assets***). MS-1004, 7:7-27; MS-1003, ¶¶191-192.

Khandekar's ***inventory*** also includes network resources as part of the

[4] As explained in Ground 3, *infra*, claim 9 lacks entitled to priority because the language recited in at least [9.1], [9.6], [9.7], and [9.8] are not found in the original claims or within its disclosure.

virtualization environment. Specifically, host hardware includes "any conventional network interface device such as one or more cards 150" to "enable communication and data transfer with external systems via a network 700." MS-1004, 6:1-12. Additionally, each VM provisioned by the system is "provided with one network card in a bridged mode" such that "[t]he VM thus also becomes available on the LAN as a machine that can be reached from any other machine in the LAN." *Id*., 14:65-15:14. The user may also "request the provisioning server to configure additional network cards and/or change the networking to be host-only." *Id*., 15:1-14. It would have been obvious that the tracked and configurable network interface resources (e.g., physical and virtual network cards) are examples of ***network resources*** included within the ***inventory*** of the ***virtualization environment***. MS-1003, ¶¶193-194.

[9.3]

As noted in [1.1] and [1.4], the provisioning server dynamically maintains an inventory in the database, which serves as a "central repository of data required by the various components." MS-1004, 18:2-5. The database includes data structures for "hosts" (e.g., tracking the "hardware configuration of the host (e.g., memory size, disk space, etc.)"), "Staging VMs" and "Instantly deployable VMs" (e.g., tracking their OS, applications, and storage locations), and "Applications" (e.g., tracking available application software). *Id*., 18:6-41. The heuristics engine

"takes this user input and applies heuristics/rules to select the most appropriate VM from a library 830 of staged VMs" and "also selects which host machine 1000, 1001 is most appropriate to provision and deploy the VM on." *Id*., 10:27-42. To make these selections, the heuristics engine scans "the list of available host machines" and determines "the one with enough memory and disk space to host the VM about to be created" by "querying the host via an API provided to manage VMs." *Id*., 20:54-58; MS-1003, ¶¶195-198.

Additionally, "[i]nformation about the current state of the different hosts is preferably input dynamically into a database or other data structure in the heuristics engine 880," and this "host state information can then be used to suggest components, as well as the properties of VMs that need to be matched with user requests." MS-1004, 13:1-8. A POSITA would have understood and found obvious that the heuristics engine and database (which collectively track host configurations, VM availability, application information, and dynamically-updated host state) are an example of the ***virtual mapping layer*** executed by the provisioning server (***control center computer***) ***that determines an inventory of network resources and virtual assets available to the virtualization environment***. This is because they determine where and how VMs can be allocated to available host resources, which functionally maps the relationship between virtual assets and physical resources within the virtualization environment. MS-1003, ¶198.

[9.4]

As noted in [9.3], the database maintains data structures that track hosts, VMs, and applications across a range of configurations. The database catalogs VMs with "different operating systems such as Linux Red Hat, Windows 2000, Windows Me, etc., and/or different versions or releases of a single operating system." MS-1004, 11:36-49. The applications library similarly contains "code for any collection of the myriad of existing applications, from web servers, database servers, spreadsheets and word-processing programs, to games and browsers." MS-1004, 11:50-58. The heuristics engine selects hosts based on resource availability (e.g., scanning "the list of available host machines" to "determine the one with enough memory and disk space") rather than based on platform-specific characteristics. MS-1004, 20:54-58. In this way, the inventory determination operates across different OSs and applications without being constrained to any particular one. MS-1003, ¶200.

Khandekar further supports that its provisioning architecture operates independently of hardware and virtualization platforms since "as long as the VMM on each host exports a known hardware interface to the VM deployed on it, then VM deployment will be substantially independent of the actual physical hardware configuration." MS-1004, 13:17-35. Additionally, the VMM's resource mapping "is transparent to the VM, and that does not require modifications to either the

guest OS or to the underlying host operating system," allowing VMs to "migrate freely, even to other hardware and software platforms." MS-1004, 8:20-35. A POSITA would have understood and found obvious that, in a system where VM deployment is designed to be "substantially independent of the actual physical hardware configuration," the heuristics engine 880 and database 890 (***the virtual mapping layer***) would have determined the inventory of available hosts, VMs, and applications without ***regard to***[5] ***processors, operating systems, virtualization platforms, and application software of the virtualization environment***. MS-1003, ¶¶201-203.

[9.5]

As noted in [9.3], the heuristics engine and database collectively function as a ***virtual mapping layer*** that determines the inventory of available hosts, VMs, and applications. The provisioning engine 810 is "the component that ties together the various components" of the system and takes "each user request and runs the

[5] Under *Wellman*, there is no need to resolve whether the portion of [9.4] reciting "without regard to processors, operating systems, virtualization platforms, and application software of the virtualization environment" is entitled to any patentable weight based on that feature being directed to what is ***not*** being "regarded.", as the prior art renders [9.4] obvious for reasons noted. MS-1003, ¶203.

corresponding provisioning operation as a separate process." MS-1004, 21:26-31. In executing each provisioning operation, the provisioning engine communicates with the heuristics engine, e.g., in a provisioning algorithm that includes the step to "[c]all the heuristics engine 880, to determine the VM's hardware configuration and the host to deploy on." *Id*., 21:33-34. The heuristics engine in turn "takes this user input and applies heuristics/rules to select the most appropriate VM from a library 830 of staged VMs" and "also selects which host machine 1000, 1001 is most appropriate to provision and deploy the VM on" by matching user input "against the database." MS-1004, 10:27-42, 20:32-42; MS-1003, ¶¶204-206.

A POSITA would have understood and found obvious that the provisioning server (***control center computer***) executes the provisioning engine 810 (***provisioning manager***) ***that communicates*** with the heuristics engine and database 890 (collectively ***virtual mapping layer***) to execute provisioning requests. MS-1003, ¶¶207-208. The provisioning engine also coordinates with processors in the hosts to execute each provisioning operation. MS-1004, 21:26-64. In this way, the provisioning engine orchestrates a sequence of operations that spans both the provisioning server (***control center computer***) and the hosts (***computer devices***), with each contributing processing steps for completing the provisioning workflow. MS-1003, ¶¶209-210.

[9.6], [9.7], [9.8]

The operations recited in these limitations correspond to those recited in [1.3], [1.5], [1.6], and [1.7] and addressed above in relation Khandekar's provisioning server. It would have been obvious to configure each of, and thus, together, both a processor of the provisioning server and a processor of the host to perform these operations. MS-1004, 9:46-10:14, 14:26-64, 20:64-67; MS-1003, ¶¶211-217.

Read in light of the '050 patent's specification, reference to the "processor" configuration in [9.6] and the corresponding operations in [9.6], [9.7], and [9.8] reflects a complement of processing capabilities distributed across the ***computer device*** and the ***control center computer***. *See* [9.1]; MS-1003, ¶¶212-216. The operative steps recited in [9.6]-[9.8] are drafted with sufficient breadth that they would have been obvious at each of the host and provisioning processors, as those coupled processors operate in coordination. *Id*. Indeed, a POSITA would have recognized that, within the same workflow the host's processor performs operations (e.g., receiving VM configuration data, registering and activating the VM on its local resources, making the VM available for user access) and that those host operations independently fall within the scope of [9.6]-[9.8], notwithstanding distinctions that may exist between those host-side operations and the inspiring server-side orchestration steps. *Id*. The prosecution history of the '050 patent

corroborates this point, as applicant's submission of amendments to claim 9 attributed the operations now recited in [9.6], [9.7], and [9.8] to the host's "processor," and for this amendment, applicant relies on the specification's limited disclosure of those operations being performed by a central computer system; the specification does not describe those steps performed by the processor of a host coupled thereto. *See id.*, ¶¶214-215.

As to Khandekar, its provisioning server and its host operate as part of a distributed computing system in which provisioning operations involve coordinated activity between both components. A host is configured to execute complementary operations of those orchestrated by the provisioning server, and in doing so, the host's processor performs operations initiated by and coordinated through the provisioning server. MS-1004, 21:50-60. Thus, initiation of the operations recited in [9.6], [9.7], and [9.8] by the provisioning server (as mapped to [1.3], [1.5], [1.6], and [1.7]) would have resulted in the host's processor being configured to perform corresponding operations as part of the same provisioning workflow. MS-1004, 10:27-42, 14:26-49, 20:43-62, 21:26-64; MS-1003, ¶217. This understanding is further corroborated by prior art confirming that a POSITA would have recognized Khandekar's provisioning architecture as involving "server-to-host delegation as an instance of this well-known pattern," as such, that the POSITA would have found it obvious to "configure the host's processor to

perform provisioning-related operations within the coordinated workflow described in Khandekar." MS-1003, ¶219; *see id.*, ¶218 (citing MS-1005, [0088] as an example).

Obviousness of the configurations discussed above are even reinforced by the fact that allocation of processing responsibilities between servers (e.g., provisioning servers) and managed hosts represented a well-understood design choice in virtualization environments by the Critical Date. MS-1003, ¶¶220-222 (citing MS-1005, [0079], [0107], [0109], [0115]). Indeed, Khandekar reveals as much when describing distribution of provisioning several operations to its host (e.g., registering the VM through a local API, executing configuration scripts, installing applications). Hirschfeld and Bandhole each confirm that this type of server-to-host delegation reflects one of a limited number of recognized architectural choices for where provisioning operations may be executed, further supporting that "the boundary between what is executed at the provisioning server versus at the host is a design choice, not a fixed architectural constraint." MS-1003, ¶¶220-222 (citing MS-1006, 4:15-28, 13:5-23; MS-1007, [0068]-[0070] for corroboration). In this way, a POSITA would have understood Khandekar's hosts as configured to perform provisioning-related operations, consistent with those performed by the provisioning server. MS-1003, ¶¶223-232. A POSITA therefore would have found it obvious to implement Khandekar such that a host's processor

is configured to independently perform provisioning-related workflows (including those recited in [9.6], [9.7], [9.8]) distinct from the provisioning server. *Id*.

3. *Claims 11, 12, 14, 16*

[11]

As noted in [3] and [9.5], the provisioning server 800 includes a provisioning engine 810 (***provisioning manager***) that provides a "user console 801" (***user interface***) that is "preferably a web server" presenting dialogs for submitting configuration options and provision requests. MS-1004, 9:46-64, 13:66-14:3, 14:26-29. Khandekar's provisioning manager would have allowed a user to log in and submit requests to the provisioning server through the ***user interface***. MS-1003, ¶¶233-234.

[12]

Khandekar's provisioning engine 810 (***provisioning manager***) provides a user console 801 (***user interface***) that allows a user to ***access*** available VM images or templates (***one or more images***) stored in a repository (e.g., "repository 840" or "library of disk drive images") (***virtual asset image repository***). MS-1004, 10:15-26, 14:4-18, 15:62-67, 17:31-49, 18:14-15; *see supra* [4], [11]; MS-1003, ¶¶235-236.

[14]

As noted in [6] and [9.4], Khandekar's ***inventory*** is maintained by the

heuristics engine 880 and database 890. This ***inventory*** includes data structures for "Applications" that index an "Applications Library 850," which is "a repository of the software and data needed to install an application 856 in a VM." MS-1004, 18:24-26, 18:57-19:2. By tracking these available software applications, tracked data relating to available VMs and network resources includes ***one or more application programs***. MS-1003, ¶¶237-238.

[16]

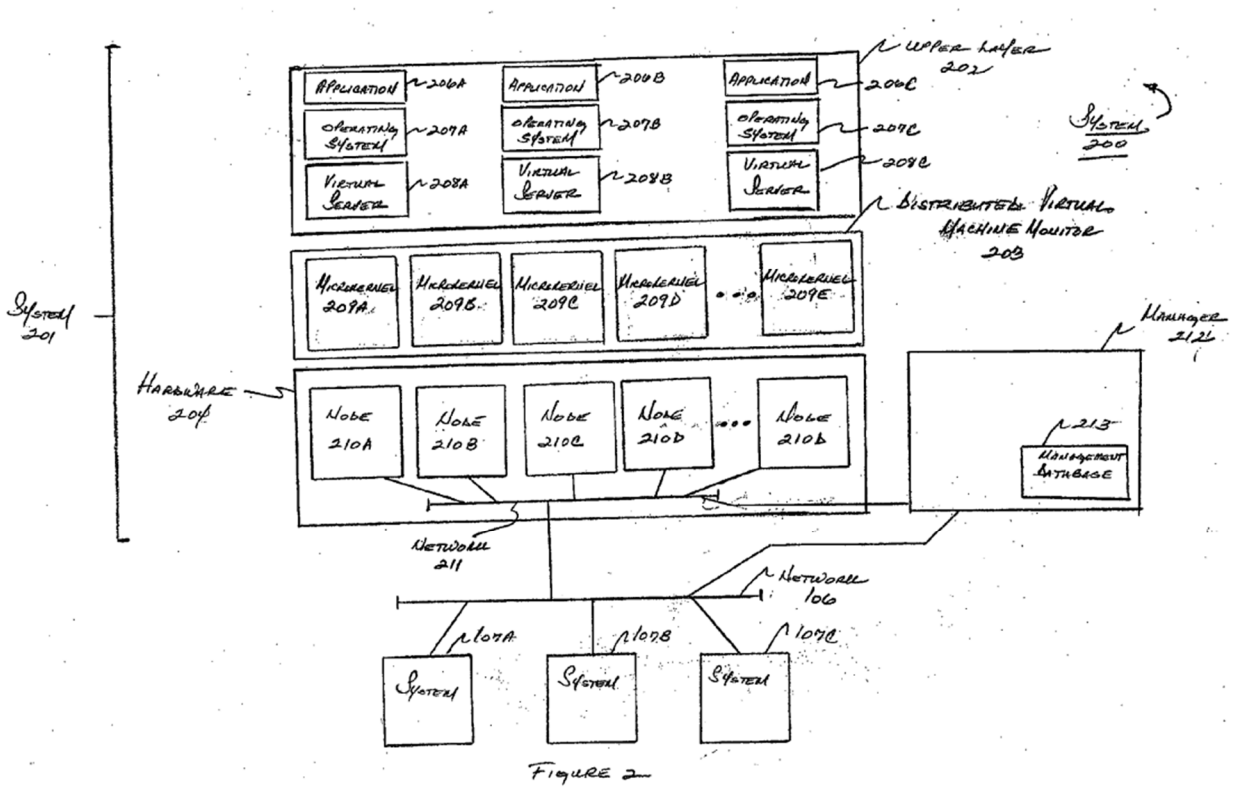
As noted in [8] and [14], Khandekar's inventory (maintained by the heuristics engine 880 and database 890) tracks an Applications Library 850 containing various software applications (***application programs***). MS-1004, 8:24-26, 18:57-19:2. Khandekar's library "contain[s] code for any collection of the myriad of existing applications, from web servers, database servers, spreadsheets…" (***network server***). MS-1004, 11:50-58. Khandekar also describes provisioning a network of computers where "a developer wants to test his web application, which uses a web server on one computer, an application server on another…" *Id.*, 23:11-20. By tracking these specific server applications as part of the overall resource pool, a POSITA would have understood and found obvious that a tracked "web server" or "database server" is an example of a ***network server***. MS-1003, ¶¶239-240.

C. GROUND 1C

1. Khandekar-Vasilevsky

(a) Vasilevsky

Vasilevsky's virtualization architecture facilitates dynamic resource allocation and management for executing virtual computing systems. MS-1005, [0070], [0072], FIG. 2 (reproduced below).



MS-1005, FIG. 2

As shown above, system 201 includes a manager 212 that manages the configuration of the system. MS-1005, [0070]. The manager 212 includes an associated management database 213 that "stores information relating to the configuration of system 201." *Id.* Manager 212 communicates with "management

agent" (not shown) that is "executed by one or more virtual servers" for "the purpose of performing configuration changes, monitoring performance, and performing other administrative functions associated with system 201." *Id.* This management architecture (including manager, management database, management agent) enables, for instance, dynamic resource management for virtual servers, including "changing the amount and allocation of resources, both actual and virtual…" *Id.*, [0072]; MS-1003, ¶¶96-98.

(b) Combination

A POSITA would have predictably combined Khandekar's VM provisioning environment with Vasilevsky's techniques for dynamic resource monitoring and allocation for virtualized resources. MS-1003, ¶¶105-106. 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 communicates with management agents "for the purpose of performing configuration changes, monitoring performance, and performing other administrative functions." MS-1005, [0070]. In the combination, Khandekar's provisioning server environment would have incorporated Vasilevsky's management capabilities (e.g., manager) so that performance-related state information may be collected and used as part

coordinating the provisioning of VM instances. 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 look for techniques that (i) make physical-to-virtual resource relationships explicit, (ii) monitor real-time performance metrics, and (iii) permit flexible allocation/reallocation of resources during operation, including through UI/CLI/API-driven management. MS-1003, ¶107. This is because Khandekar makes clear that its provisioning workflows rely on dynamically-maintained resource state information. Specifically, "[i]nformation about the current state of the different hosts is preferably input dynamically into a database" used by its heuristics engine and the heuristics engine can "query[] the host via an API" and "[r]equest the host (using the API) to reserve the amount of memory and disk space required." MS-1004, 12:57-13:8, 20:54-60.

A POSITA would have understood that when utilizing dynamic state tracking and API-driven resource reservation during provisioning, there is a predictable need for administrable mechanisms for monitoring resource allocation as conditions change (e.g., during operation and across many provisioned assets). MS-1003, ¶108. This would have led the POSITA to Vasilevsky, which identifies a limitation of conventional approaches and teaches provisioning in real-

time without a restart of a virtual computing system. MS-1005, [0076]; MS-1003, ¶108.

A POSITA implementing Khandekar's provisioning workflow would have been motivated to incorporate known virtualization management techniques that provide clearer visibility in how resources are consumed by virtual computing systems. MS-1004, 10:27-42, 21:26-31; MS-1003, ¶109. Khandekar states provisioning decisions are resource-driven (e.g., a heuristics engine "determine[s] the [host] with enough memory and disk space," and "request[s] the host (using the API) to reserve the amount of memory and disk space required"), and that provisioning is treated as an orchestrated, transaction-like workflow whose progress is monitored and controlled through the provisioning server's management plane. MS-1004, 20:54-21:24.

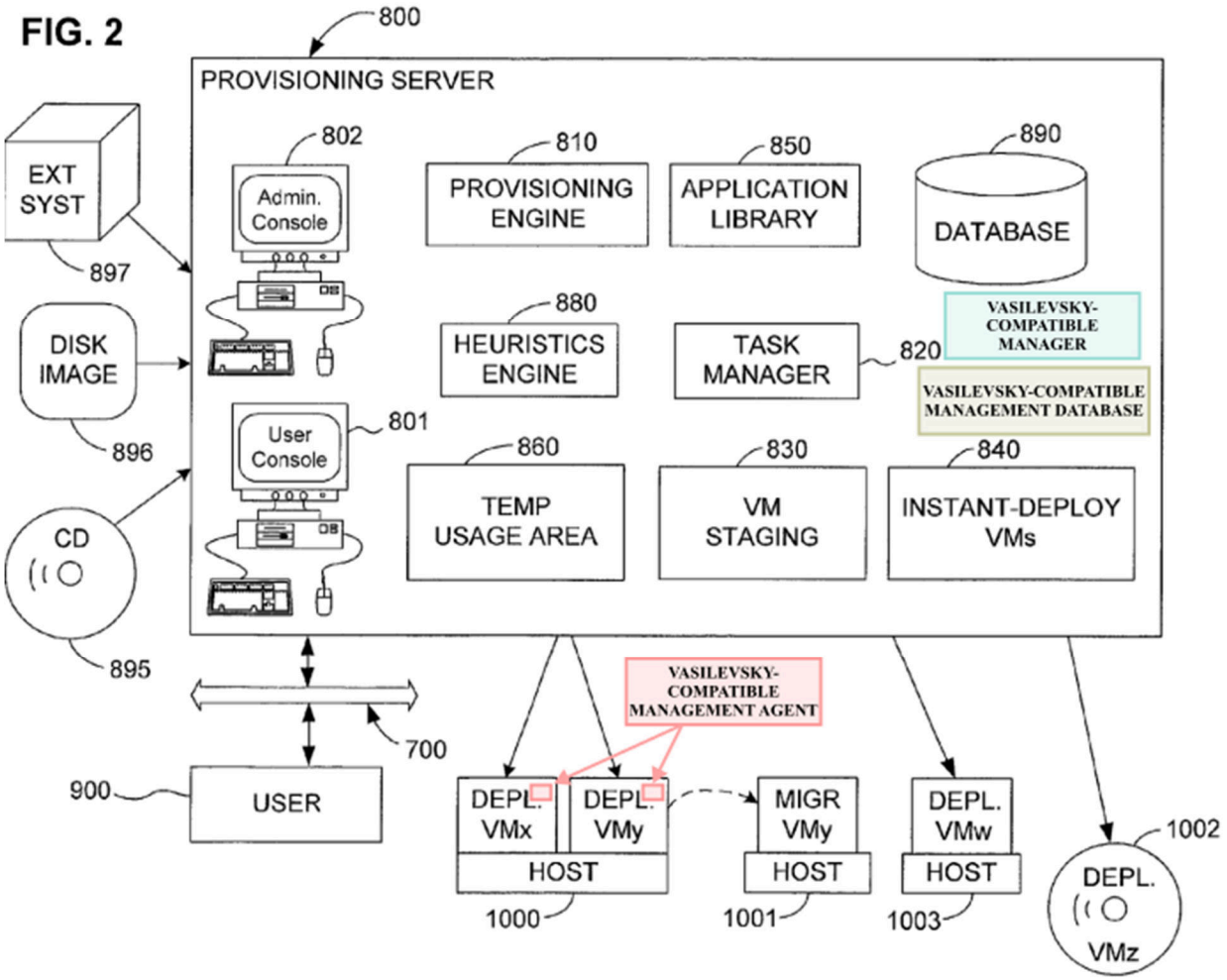
A POSITA would have been motivated to look for established approaches that improve resource monitoring for provisioned VMs. MS-1003, ¶110. A POSITA would have found Vasilevsky (which describes a management agent that monitors performance metrics and maintains a mapping database) to be pertinent to Khandekar. MS-1005, [0084], [0150]. Incorporating those teachings into the system would have been a predictable, compatible refinement to Khandekar's provisioning server (e.g., by incorporating functionality provided by Vasilevsky's manager 212 and management database 213), because it supplies a well-known

architecture for monitoring and managing resource allocations (e.g., disk/memory reservation and assignment)—allocations that Khandekar already leverages for its provisioning decisions and provisioning-related tasks. MS-1004, 9:46-64, 20:54-60, 22:53-67; MS-1003, ¶110. Moreover, to enable monitoring capabilities, each VM provisioned through Khandekar's provisioning environment would similarly incorporate functionality associated with Vasilevsky's management agent, thereby enabling communication with a manager within the provisioning server. MS-1003, ¶110.

In one example implementation, Khandekar's provisioning server would have included (or operated in conjunction with) a Vasilevsky-compatible manager (e.g., manager 212) and associated management database (e.g., management database 213). In this example, management database stores and manages configuration and monitoring information for virtualized instances. MS-1004, 9:26-32; MS-1005, [0070]. One or more management agents (executed by provisioned VMs) communicate with the manager "for the purpose of performing configuration changes, [and] monitoring performance," thereby enabling the collection of performance and loading information associated with provisioned VMs. MS-1005, [0070], [0072]; MS-1003, ¶111.

In Khandekar-Vasilevsky, a user would continue to submit a provisioning request through Khandekar's user console 801 (e.g., by selecting an "instantly

deployable" VM). MS-1004, 9:65-10:14, 13:66-14:3. The provisioning server would select an appropriate host and deploy the requested VM on that host (as described in Khandekar), including using the types of information described in Vasilevsky (e.g., dynamic resource allocation). MS-1004, 10:27-42, 12:57-13:8, 20:54-60; MS-1005, [0072]. After deployment, deployed VMs and hosts "can be monitored," including monitoring hosts for "load," and the combined system would use the Vasilevsky-compatible management (and communications with management agents) to receive monitored "performance" and "loading" information for the executing VMs and/or their supporting hosts. MS-1004, 13:44-56; MS-1005, [0070], [0072]; MS-1003, ¶112.



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

Incorporating Vasilevsky's teachings would have reflected an implementation for representing and controlling how virtualized constructs correspond to underlying physical resources (including during execution), while preserving Khandekar's provisioning-server control plane and provisioning tasks. MS-1003, ¶113. A POSITA would have also reasonably expected success in doing so at least because the combination amounts to using known

virtualization-management techniques (physical-to-virtual mapping, programmatic allocation) to enhance a known VM provisioning workflow that already operates through well-defined management interfaces to hosts and already contemplates computer-generated provisioning requests. *Id.*

2. **Analysis**

(a) Claim 2

[2]

As noted in [1.5], Khandekar's VMs (***provisioned virtual assets***) are deployed based on a user's provisioning request. Once deployed, "[t]he deployed VMs can be monitored regularly" and that "[h]osts can be monitored for load." MS-1004, 13:44-56. Khandekar-Vasilevsky would have further incorporated Vasilevsky's management agent architecture to enhance Khandekar's provisioning workflow by providing clearer visibility and runtime control over how these virtual resources consume physical resources. Vasilevsky's management architecture communicates with a management agent executing on the system "for the purpose of performing configuration changes, [and] monitoring performance." MS-1005, [0070]. Resources are evaluated and adjusted based on the changing requirements of the applications or operating systems, including "performance, loading, etc." *Id.*, [0072]. Thus, by actively monitoring performance and loading data points (***one or more performance metrics***) for VMs, Khandekar-Vasilevsky would have

determined one or more performance metrics of the provisioned VMs (***provisioned virtual asset***). MS-1003, ¶¶241-242.

(b) Claim 5

[5]

As noted in [2], Khandekar-Vasilevsky would have actively monitored performance and loading data for the provisioned VMs. To enable runtime adjustments, Vasilevsky's system responds to these changing requirements by resizing the virtual server "automatically (e.g., dynamically in real time)." MS-1005, [0014]. The management architecture also enables provisioning of computing resources "in real-time ('on-the-fly') without a restart of a virtual computing system." *Id.*, [0077]. A POSITA would have understood and found obvious that to resize virtual servers dynamically in real time based on changing loading and performance conditions, the system would have evaluated those underlying metrics in real time. MS-1003, ¶¶243-244. Thus, by monitoring and utilizing performance data to execute "on-the-fly" resource adjustments, Khandekar-Vasilevsky would have ***determined one or more real time performance metrics***. *Id*.

(c) Claim 10

See [2], *supra*; MS-1004, 13:44-56; MS-1005, [0070], [0072]; MS-1003, ¶¶245-246.

(d) Claim 13

See [5], *supra*; MS-1005, [0014], [0077]; MS-1003, ¶¶247-248.

D. GROUND 1D

1. Khandekar-Tremain

(a) Tremain

Tremain describes providing computer services to multiple customers using isolated virtual machines hosted on shared physical computers. MS-1009, Abst. Tremain recognizes that hosting multiple organizations on a server creates "security, manageability and availability issues," which creates a need for "enhanced security isolation" to ensure that virtual servers and their data are "held isolated from each other." *Id.*, 4:6-19, 27:1-26. To address these vulnerabilities, Tremain describes deploying "[e]mbedded 'virtual security firewalls'" to provide protection to the virtual servers and networks. *Id.*, 32:7-11. These virtual security firewalls also consist of a "special firewall software application" and are "preferably implemented by a virtual machine on the real computer… running firewall software." *Id.*, 32:13-33:10; MS-1003, ¶¶99-102.

(b) Combination

Khandekar teaches a provisioning workflow in which a user selects a desired VM configuration and submits a provisioning request to a provisioning server. MS-1004, 9:65-10:14, 13:66-14:3. This customization allows the user to select

specific software from an "Applications Library 850" to be installed unattended on the newly provisioned VM. *Id.*, 18:57-19:2. Khandekar also contemplates certain deployments in which "more than one virtual machine can normally be installed on a single host server," and emphasizes isolation and access control, including that VMs should be "fully isolated from other virtual machines installed on the same host" to ensure "there will be no loss of security," and that network exposure may be controlled (e.g., whether a VM "can be reached from any other machine in the LAN"). MS-1004, 5:37-49, 15:1-14; MS-1003, ¶¶114-115.

In this context, Tremain provides a straightforward example of one such security-oriented application program, e.g., a software-based firewall implemented within a virtual machine. MS-1003, ¶¶116-118. Like Khandekar, Tremain recognizes that multi-tenant configurations raise "security, manageability and availability issues." MS-1009, 4:6-19. To address these issues, Tremain proposes using "[e]mbedded 'virtual security firewalls'" that "provide protection to virtual servers and virtual networks." *Id.*, 32:7-11. The firewall "consist[s] of a special firewall software application" and is "preferably implemented by a virtual machine on the real computer" that is "running firewall software." MS-1009, 4:6-19, 27:1-26, 32:7-11, 32:13-33:10.

A POSITA would have incorporated Tremain's firewall into Khandekar's existing Applications Library workflow by including firewall software as an

available application program (to be selected and installed during provisioning). This would have been a predictable combination that improves security isolation in Khandekar's shared-host deployments while preserving Khandekar's core automated provisioning workflow. MS-1004, 11:50-58, 18:57-19:2; MS-1009, 32:13-33:10; MS-1003, ¶¶119-120.

2. **Analysis**

(a) Claim 7

As discussed in Ground 1 for [6], Khandekar's database 890 includes an Applications Library 850 containing various software applications (***application programs***). MS-1004, 18:24-30, 18:57-19:2. This library "contain[s] code for any collection of the myriad of existing applications" that can be selected and installed onto a provisioned VM. *Id.*, 11:50-58. Further, Khandekar describes that a VM may run an application representing an "database server" thereby enabling access to a ***storage server***. MS-1004, 11:50-58; *see id.*, 23:11-20; MS-1003, ¶¶249-250.

A POSITA would have found it obvious to implement Khandekar's applications library based on the teachings of Tremain to include ***firewall*** software within the applications library including the ***application programs***. MS-1003, ¶¶249-251. Such a configuration would have enhanced the security and isolation of VMs provisioned using Khandekar's provisioning techniques. *Id*. Tremain describes that "virtual security firewalls operate similarly to conventional firewalls

in that they consist of a special firewall software application which runs on a host operating system on a computer." MS-1009, 32:13-33:10. Such a firewall "is preferably implemented by a virtual machine on the real computer… running firewall software." *Id.*, 17:16-19.

(b) Claim 15

See [7], *supra*; MS-1004, 11:50-58, 18:24-30, 18:57-19:2; MS-1009, 17:16-19, 32:13-33:10; MS-1003, ¶¶252-254.

E. GROUND 2A

1. 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.

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.

Accordingly, 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, 6:28-7:30.

Hirschfeld also 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, 17:9-24; MS-1003, ¶¶85-95.

2. **Analysis**

(a) Claim 1

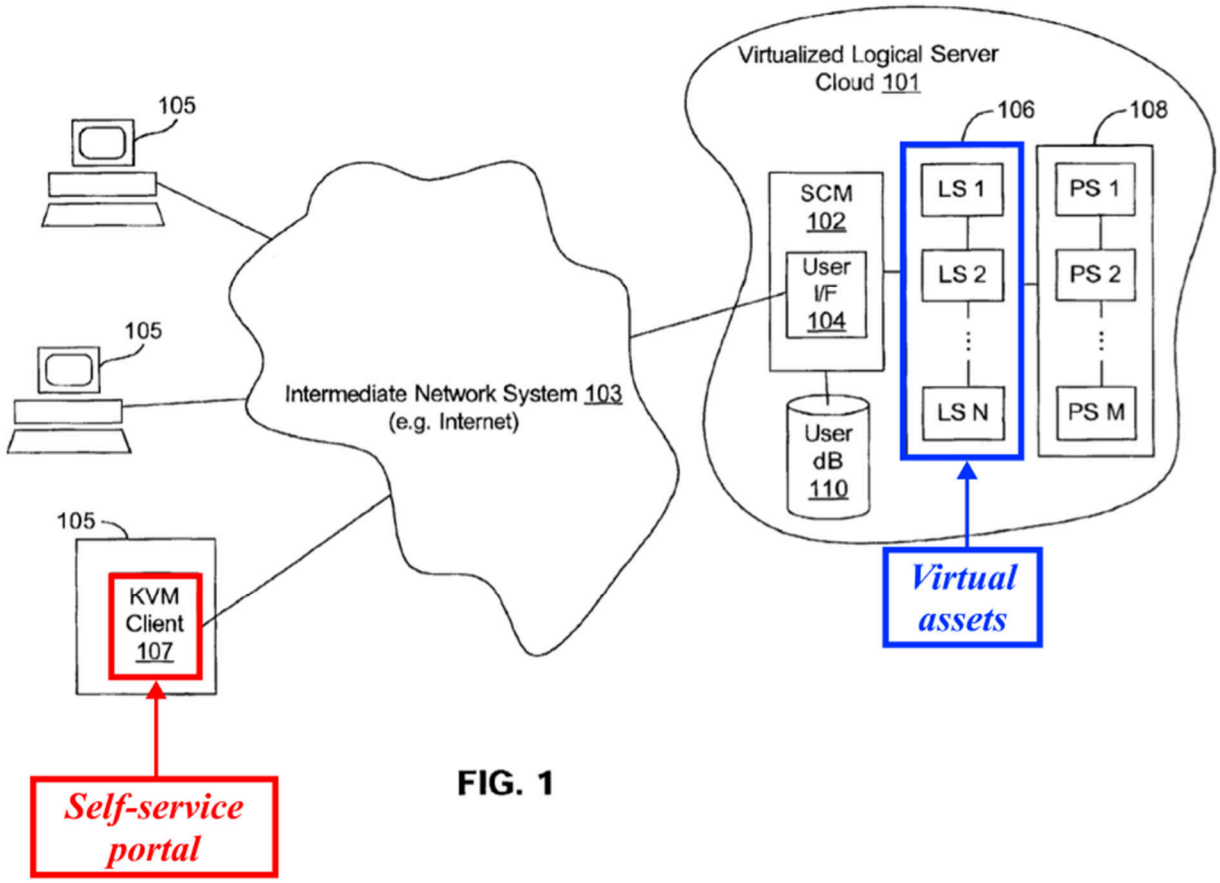
[1.pre]

To the extent the preamble is limiting, Hirschfeld renders [1.pre] obvious. MS-1003, ¶255-262. 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. Hirschfeld's SCM provides a "web page portal or other access mechanism" through which "a user of any remote terminal" may "login to and

access any of the logical servers" of the virtualized logical server cloud via a user interface and the SCM. MS-1006, 6:28-63, 6:64-7:30; MS-1003, ¶257-259. The user interface "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, 7:22-30. In this configuration, the web page portal (user interface) provides an example of a ***self-service portal*** through which the user may select and interact with logical servers that exist only in a virtualization environment (***virtual assets***) and initiate instancing functions (e.g., copy, move) under SCM control (***on demand provisioning***). MS-1006, 6:64-7:30, 7:22-30; MS-1003, ¶¶258-259.

The SCM also coordinates logical server instancing procedures, including replication of a logical server on a physical server by retrieving and using a logical server template and activating a logical server from the resulting configuration information. MS-1006, 4:15-28. These instancing procedures involve associating a logical server (***virtual asset***) with a particular physical server, which is an example of the ***provisioning*** recited in the claim. MS-1003, ¶260. At least because the user interface provides access to instancing functions (e.g., copy, move) (MS-1006, 7:22-30), a POSITA would have understood and found obvious that a user would have initiated such ***provisioning*** through the SCM's web page

portal on demand. MS-1003, ¶260-262.



MS-1006, FIG. 1.

[1.1]

As noted in [1.pre], Hirschfeld's system (***virtualization system***) provides a "virtualized logical server cloud" that includes a plurality of logical servers, a plurality of physical servers, and a server cloud manager. MS-1006, 2:63-3:11. Each physical server is coupled to physical resources including a "network resource," a "data storage resource" and a "processor resource." *Id.* Each physical server executes virtualization software that virtualizes physical resources for

logical servers linked to that physical server. *Id.*; *see also id.*, 9:35-57. Thus, in Hirschfeld, physical servers with resources (e.g., network, storage, processor) (***physical resources***) and logical servers (***virtual assets***) are managed by a virtualization layer running on the physical resources (***virtualization software executing on the physical servers***). MS-1006, 2:63-3:11, 9:35-57; MS-1003, ¶¶263-270.

The SCM establishes and maintains "status and instance information" for the logical servers, such as attributes that link each logical server with a physical server. MS-1006, 2:63-3:20. The SCM also maintains a user database that includes "mapping information between users, the logical servers and the physical servers." *Id*. Additionally, resource allocation information is tracked for each logical server, including non-persistent attributes. Examples include processor resources (e.g., "the percentage of CPU time allocated to a particular logical server") and memory resources (e.g., "the amount of memory available to a logical server"). MS-1006, 12:21-43. The SCM may also "redistribute processor resources based on demand and usage." MS-1006, 12:21-43. A POSITA would have understood and found obvious that tracked information (e.g., maintained status/instance records, mapping information, and resource allocation data) is an example of ***inventory*** of the ***physical resources*** and ***virtual assets*** available within the virtualization system and their associations. MS-1003, ¶¶268-270.

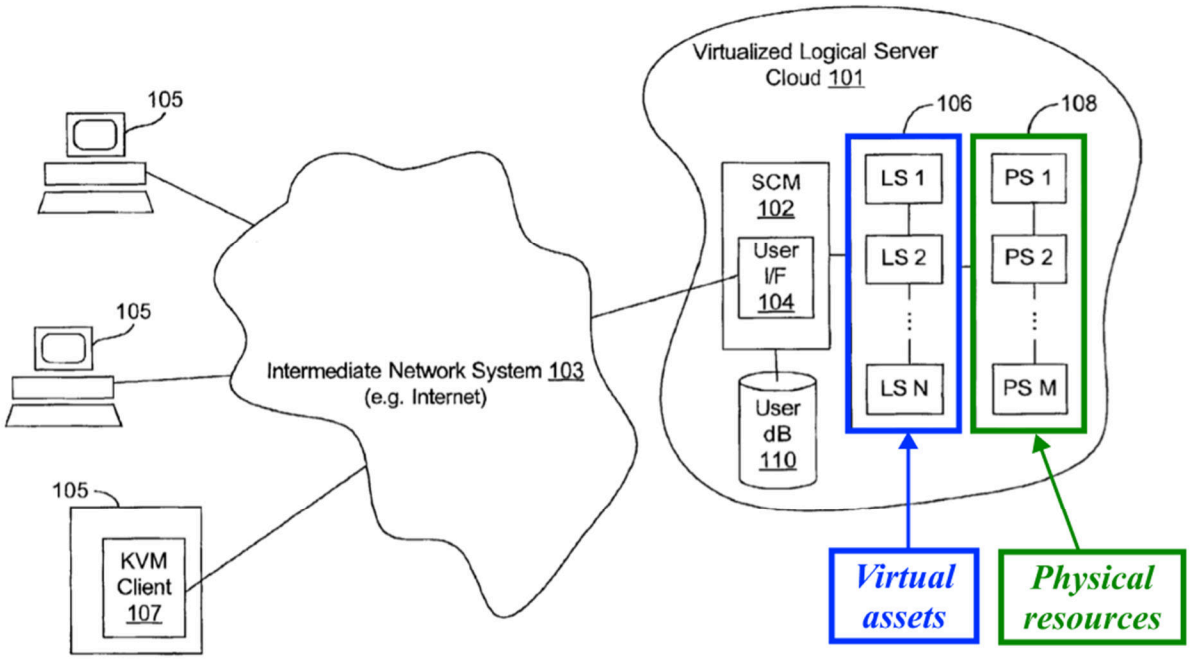


FIG. 1

MS-1006, FIG. 1.

[1.2]

As noted in [1.1] (Ground 2A), Hirschfeld describes the use of "virtualization technology that enables multiple logical servers to operate on a single physical computer," including virtualization that allows multiple logical servers to "override a physical server's attributes and share its resources." MS-1006, 1:63-2:7. Each logical server includes a "hardware" layer, an operating system layer, and a software layer. MS-1006, 10:9-27, 10:32-56. The logical server "hardware" layer provides virtualized resources (e.g., virtualized network, storage, memory, and processor resources) corresponding to physical resources of

the underlying physical server. *Id.* Thus, Hirschfeld's logical servers (***virtual assets***) are implemented in software as virtualized computer systems, i.e., ***software instantiations of computer systems***. MS-1003, ¶¶271-274.

[1.3]

As noted in [1.pre], Hirschfeld's SCM provides a "web page portal or other access mechanism." A user uses a "local browser to access a web page" and interacts with the SCM via the user interface. MS-1006, 6:28-63, 6:64-7:30. The user interface "provide[s] 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." *Id.*, 7:22-30. When a user initiates, for example, an instancing function (e.g., move, copy) for a particular logical server through the portal, the request represents an example of a ***virtual asset provisioning request*** (e.g., a request to associate a logical server with a physical server under SCM control). *Id.*; MS-1003, ¶¶275-276.

The user's request identifies the logical server to be instanced (***parameters to be considered for the provisioning request***). More specifically, each logical server has persistent attributes that "establish its identity" ("a media access control (MAC) address, a boot disk image, a system identifier, a processor type and access credentials") and semi-persistent attributes (e.g., "an internet protocol (IP) address, a logical name, a server cloud manager identifier, user account information, a non-

boot disk image and network connections"). MS-1006, 3:35-44; *see also id.*, 11:58-12:19. A logical server may be moved to any other "compatible" physical server that "provides the physical and logical resources needed to retain the persistent attributes of the logical server." *Id.*, 8:11-38. This includes physical resources (e.g., "CPU type, storage capacity, disk configuration, and network connectivity") and logical constraints (e.g., "administrative control" and "software licenses"). MS-1006, 8:11-38. A POSITA would have understood and found obvious that, by identifying a particular logical server in an instancing request, the user specifies (e.g., by reference to that logical server's persistent attributes) the parameters the SCM considers in selecting a compatible physical server for the requested association. MS-1003, ¶¶277-280.

Additionally, reallocation of a logical server may be "directed by the user in the form of an upgrade or downgrade or any other modification requested by the user." MS-1006, 17:25-39. Such user-directed modifications provide an additional example of a ***virtual asset provisioning request*** in which the user specifies parameters (e.g., desired resource levels) for the SCM to consider in associating the logical server with appropriate physical resources. MS-1006, 17:25-39, 12:45-13:3; MS-1003, ¶281.

[1.4]

A POSITA would have understood that Hirschfeld's system receives and maintains data reflecting the inventory available in the virtualized logical server cloud (***data on inventory***), including status/instance information and attribute/mapping information identifying logical servers (***virtual assets***), their associated physical servers, and user access associations. MS-1006, 2:63-3:20, 3:50-65, 11:4-25, 11:45-57, 12:45-13:3; MS-1003, ¶¶282-286.

As noted in [1.1], the SCM "establishes and maintains status and instance information" for the logical servers, including "the persistent and any non-persistent attributes that link each logical server with a physical server." MS-1006, 2:63-3:20. A user database is maintained and "includes mapping information between users, the logical servers and the physical servers." *Id.* A POSITA would have understood and found obvious that this information represents examples of data describing the inventory available in the virtualization system (e.g., which logical servers exist, their attributes/capabilities, and their linkage/associations to physical servers/resources and users). MS-1003, ¶¶283-284.

The SCM also interfaces with and controls physical server agents ("PSAs") and logical server agents ("LSAs") to manage physical and logical attributes of logical servers. This includes a PSA running on the physical server and controlling physical server functions and associated resources via interfaces to the

virtualization software. The SCM "manages all of the LSAs" and PSAs, "maintains the status and instance information for each logical server," and issues instructions to physical servers via corresponding PSAs. MS-1006, 3:50-65, 11:4-25, 11:45-57, 12:45-13:3. SCM's maintenance of status/instance information and mappings is implemented through ongoing data exchange with these controlled agents and/or the underlying servers, thereby reflecting a receipt of examples of ***data on inventory*** available in the virtualization system. MS-1003, ¶¶285-286.

[1.5]

As noted in [1.3], in Hirschfeld, a logical server may be moved to any other "compatible" physical server, and a compatible physical server is one that "provides the physical and logical resources needed to retain the persistent attributes of the logical server." MS-1006, 8:11-38. This includes physical resources, such as "CPU type, storage capacity, disk configuration, and network connectivity," as well as logical constraints such as "administrative control" and "software licenses." *Id*. Because the user's provisioning request identifies a particular logical server whose persistent attributes define the applicable resource and compatibility requirements (as noted in [1.3]), the SCM's selection of a compatible physical server (***particular physical resource***) that satisfies those requirements reflects an example of ***provisioning*** of the logical server (***particular virtual asset***) to a physical resource that matches the parameters in the

provisioning request. MS-1003, ¶¶287-289.

The SCM also automatically orchestrates procedures to associate logical servers with physical servers, including move procedures in which the SCM copies a disk image from a first physical server to a second physical server, copies logical server configuration, and remaps [keyboard/video/mouse (KVM)] and network functions to the target physical server. MS-1006, 3:66-4:13, 13:5-23. The SCM also performs replication procedures, where a physical server agent is instructed to "retrieve[] and locally store[] a logical server template," "creat[e] logical server configuration information from the logical server template," and "activat[e] a logical server from the logical server configuration information." *Id*., 4:15-28. The SCM also "maintains the status and instance information for each logical server" and "issues instructions to each physical server via corresponding PSAs" to execute these operations. *Id*., 11:45-57. These configurations involve examples of ***provisioning*** operations (e.g., associating a logical server with a compatible physical server) and are performed automatically under SCM control on selected physical server resources that satisfy the applicable compatibility requirements (i.e., match the parameters specified in the provisioning request). MS-1003, ¶¶290-294.

[1.6]

Hirschfeld's system ***assign[s]*** the provisioned logical server (***provisioned virtual asset***) to the corresponding user-initiated provisioning request (***virtual asset provisioning request***) by recording and maintaining, within SCM-managed data structures, the status/instance information and mapping/association information for the logical server. MS-1006, 2:63-3:20, 7:32-62, 11:45-57, 12:45-13:3, 14:1-17; MS-1003, ¶¶295-301. This includes an updated association to the selected physical server and the requesting user.

As noted in [1.1] and [1.4], the SCM "establishes and maintains status and instance information for the plurality of logical servers including the persistent and any non-persistent attributes that link each logical server with a physical server." MS-1006, 2:63-3:11. The SCM tracks logical server configuration information in a local database, such as when a logical server is moved or recreated on a target physical server. The database tracking the logical server configuration information "is reconfigured, such as modifying links or references, to track the new LS configuration information" on the target physical server. *Id.*, 14:1-17. The SCM also "maintains the status and instance information for each logical server" on an ongoing basis and "issues instructions to each physical server via corresponding PSAs." *Id*., 11:45-57.

Further, a user database "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. The SCM "automatically accesses the appropriate physical server corresponding to the selected logical server." MS-1006, 7:32-48. When the SCM performs a ***provisioning*** operation (e.g., move, replication) to associate a logical server with a physical server in response to a user's request, the SCM's reconfiguration of its database tracking (to reflect the logical server's new physical server linkage) and the user database's continued mapping of the user to that logical server together operate to assign the provisioned virtual asset to the provisioning request and the corresponding user. MS-1006, 7:32-48, 14:1-17; MS-1003, ¶¶299-301.

[1.7]

Hirschfeld ***provide[s] user access to the provisioned*** logical server (***virtual asset***) through the SCM's user interface (e.g., a web page portal) and associated access mechanisms (e.g., KVM redirection/mapping). This enables a user of a remote terminal to login and access logical servers for which the user has permission without needing to know physical server access information. MS-1006, 6:28-63, 6:64-7:30, 7:32-62, 14:39-61; MS-1003, ¶¶302-307.

As noted in [1.pre], the SCM provides a "web page portal or other access mechanism" through which "a user of any remote terminal" may "login to and

access any of the logical servers" of the virtualized logical server cloud via the user interface and SCM. MS-1006, 6:28-63. The user database "incorporates a mapping of users, the logical servers and the physical servers," (including "link information that maps each logical server with one of the physical servers") such that the SCM "automatically accesses the appropriate physical server corresponding to the selected logical server." In this regard, the user "does not have to supply physical server information and, in fact, does not even need to be aware of any specific physical server access or location information." MS-1006, 7:32-62; MS-1003, ¶¶303-304.

Hirschfeld also describes SCM-managed access mechanisms such as KVM redirection. In such configurations, the SCM "provides KVM redirection either by providing a KVM interface or by mapping the KVM client to allow direct KVM access to the selected logical server." MS-1006, 6:64-7:30. For example, the SCM may include a KVM service and KVM client that map "KVM functions between the KVM service and the virtual KVM function" of the physical server on which the logical server resides. *Id*., 14:39-61. These mechanisms provide the user a pathway to access and interact with the logical server after it has been provisioned to a physical server. MS-1003, ¶¶305-307. Thus, in Hirschfeld, the user is ***provid[ed] access*** to the provisioned virtual asset (***logical server***) via the SCM portal and associated access mechanisms.

(b) Claim 3

[3]

As noted in [1.pre] and [1.3], the SCM provides a "web page portal or other access mechanism" (***user interface***) through which "a user of any remote terminal" may "login to and access any of the logical servers" of the virtualized logical server cloud via a user interface and the SCM. MS-1006, 6:28-63, 6:64-7:30. The user interface "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, 7:22-30; MS-1003, ¶¶308-309.

When a user initiates an instancing function (e.g., move, copy) for a particular logical server (***virtual asset***) through the portal, the request is an example of a ***virtual asset provisioning request*** (e.g., a request to associate a logical server with a physical server under SCM control). MS-1003, ¶310. The SCM's web page portal through which the user initiates such a request thereby provides an example of a ***user interface for submitting the virtual asset provisioning request***. MS-1006, 6:28-63, 6:64-7:30, 7:22-30; MS-1003, ¶¶308-312.

F. GROUND 2B

1. Hirschfeld-Vasilevsky

A POSITA would have predictably combined Hirschfeld's virtualization

environment with Vasilevsky's techniques for dynamic resource monitoring and allocation for virtualized resources. MS-1003, ¶¶121-123; *see* §III.C.1(b). In Hirschfeld's virtualization environment, the SCM maintains status/instance information for logical servers, tracks resource allocation data (e.g., "the percentage of CPU time allocated to a particular logical server" and "the amount of memory available to a logical server"), and "may redistribute processor resources based on demand and usage." MS-1006, 2:63-3:11, 12:21-43.

As discussed in Ground 1C, Vasilevsky describes a manager (e.g., manager 212) that communicates with management agents "for the purpose of performing configuration changes, [and] monitoring performance," where resources are evaluated and adjusted based on the changing requirements of applications or operating systems, including "performance, loading, etc." MS-1005, [0070], [0072]; MS-1003, ¶123.

A POSITA implementing Hirschfeld's SCM-managed logical server cloud would have incorporated Vasilevsky's management agent architecture to enhance the SCM's resource monitoring and allocation capabilities. MS-1003, ¶¶124-125. While the SCM already tracks resource allocation information and redistributes resources based on demand and usage, Hirschfeld does not specifically address dedicated mechanisms for collecting and evaluating runtime performance data from executing logical servers. *See* MS-1006, 12:21-43; MS-1003, ¶124. In this

context, Vasilevsky's teachings provide relevant and compatible techniques for monitoring performance and loading data in real time, including the ability to resize virtual servers "automatically (e.g., dynamically in real time)" and to provision computing resources "in real-time ('on-the-fly') without a restart of a virtual computing system." MS-1005, [0014], [0072], [0077]; MS-1003, ¶124.

A POSITA would have recognized that incorporating Vasilevsky's management agent architecture into Hirschfeld's SCM framework advances Hirschfeld's existing agent-based management design (e.g., in which PSAs and LSAs already communicate with the SCM to manage physical and logical attributes of logical servers). MS-1003, ¶125-127. This may be accomplished without requiring changes to Hirschfeld's underlying provisioning and control sequence for managing logical servers on physical resources. MS-1006, 3:50-65, 11:4-25, 11:27-43, 11:45-57. A POSITA would have reasonably expected success because the combination involves applying known virtualization management techniques (performance monitoring, dynamic resource allocation) as predictable enhancements to Hirschfeld's SCM management framework, which already operates through well-defined agent-based interfaces to physical and logical servers and already contemplates resource tracking and redistribution. MS-1003, ¶128.

2. Analysis

(a) Claim 2

Hirschfeld-Vasilevsky would have ***determine[ed]*** "performance" and "loading" data (***one or more performance metrics***) of the logical servers (provisioned virtual asset) via a management architecture by actively monitoring performance and loading data points for logical servers. MS-1006, 12:21-43; MS-1005, [0070], [0072]; MS-1003, ¶¶313-317; *see* Ground 1C (addressing [2]).

(b) Claim 5

Hirschfeld-Vasilevsky would have evaluated (***determine[ed]***) the performance and loading data of the logical servers "dynamically in real time" or "in real-time" (***one or more real time performance metrics***). MS-1005, [0014], [0077]; MS-1003, ¶¶318-321; *see* Ground 1C (addressing [5]). By monitoring and utilizing performance data to execute "on-the-fly" resource adjustments, Hirschfeld-Vasilevsky would have been configured to determine one or more real time performance metrics. MS-1003, ¶322.

G. GROUND 2C

1. Hirschfeld-Tremain

Hirschfeld-Tremain would have predictably combined Hirschfeld's virtualization environment with Tremain's teaching of a software-based firewall implemented within a virtual machine. MS-1003, ¶¶129-132; *see* §III.D.1(b). In

Hirschfeld's virtualization environment, multiple logical servers are hosted on shared physical servers, where each logical server "operates substantially independent of other logical servers and provides virtual isolation among users." MS-1006, 2:1-7. Hirschfeld further recognizes that the shared network "may include a firewall or the like associated with a particular network fabric, which provides an interface to the network system." *Id*., 8:39-64. As discussed in Ground 1D, Tremain teaches that multi-tenant virtualized hosting configurations raise "security, manageability and availability issues," and proposes using "[e]mbedded 'virtual security firewalls'." MS-1009, 4:6-19, 27:1-26, 32:7-11, 32:13-33:10; MS-1003, ¶¶131-132.

A POSITA implementing Hirschfeld's multi-tenant logical server cloud would have been motivated to incorporate Tremain's virtual firewall teaching to enhance the security and isolation of logical servers provisioned and managed within Hirschfeld's SCM framework, because Hirschfeld already contemplates network-level security (e.g., a firewall associated with a network fabric) and virtual isolation among logical servers, but does not specifically teach implementing firewall functionality as a software application within a logical server. MS-1006, 2:1-7, 8:39-64; MS-1003, ¶¶130-132. In this context, Tremain's teachings provide a relevant and compatible technique for enhancing security in a shared virtualized environment by implementing firewall software as an application program running

on a virtual machine. MS-1009, 32:7-11, 32:13-33:10; MS-1003, ¶¶132-133.

A POSITA would have further recognized that incorporating Tremain's virtual firewall as an available application program within Hirschfeld's logical server cloud is complementary to Hirschfeld's existing architecture for hosting multiple logical servers with independent operating systems and applications on shared physical servers. This may be accomplished without meaningful changes to Hirschfeld's underlying SCM management and provisioning framework for managing logical servers on physical resources. MS-1006, 10:9-31, 15:23-67; MS-1003, ¶134. In this regard, a POSITA would have reasonably expected success because the combination involves applying a known security technique (software-based firewall running on a virtual machine) as a predictable application program within Hirschfeld's logical server cloud, which already supports logical servers executing independent operating systems and desired software applications. MS-1003, ¶135.

2. **Analysis**

(a) Claim 6

As noted in [1.1], Hirschfeld includes a software layer "for executing desired software applications." MS-1006, 2:63-3:20, 10:9-31. Tremain supplements Hirschfeld through disclosure that virtual servers in a multi-tenant virtualized hosting environment are configured with specific application programs to deliver

services, such as "pre-configured virtual servers" that are "virtual storage container files and include virtual machine operating systems and application and/or server software for the different customers." MS-1009, 54:4-29. These virtual servers are configured with "operating system and Web server software," virtual servers "loaded with application software," and virtual servers "loaded with database management system software." MS-1009, 70:12-28. A POSITA would have understood and found obvious that, in Hirschfeld-Tremain, the ***inventory*** maintained by the SCM would further include information about application programs (***one or more application programs***) available on or for the logical servers as this additional information would be pertinent to the SCM's contemplated purpose of managing and provisioning logical servers that deliver application services to users. MS-1003, ¶¶323-327.

(b) Claim 7

As noted in [6], in Hirschfeld-Tremain, the ***inventory*** maintained by the SCM would have included information about application programs (***one or more application programs***) available on or for the logical servers. MS-1006, 2:63-3:20, 10:9-31; MS-1009, FIGS. 2, 6; MS-1003, ¶¶328-332. A POSITA would have also found it obvious to include a ***firewall*** among the application programs available within Hirschfeld's logical server cloud based on Tremain's teachings. MS-1003, ¶¶329-332. Such a configuration would have enhanced the security and

isolation of logical servers provisioned and managed within Hirschfeld's SCM framework. *Id.* Tremain describes that "virtual security firewalls operate similarly to conventional firewalls in that they consist of a special firewall software application which runs on a host operating system on a computer." MS-1009, 32:13-33:10. Such a firewall "is preferably implemented by a virtual machine on the real computer" that is "running firewall software." *Id.*, 17:16-19; MS-1003, ¶¶329-331.

H. GROUND 3

This ground relies on Pabari and Bonazzola that qualify as prior art because claim 9 is not entitled to its proclaimed priority date. As explained below, the "processor" configuration recited in claim 9 defeats the '050 patent's claim to priority and renders claims 9-16 obvious based on the combination of Pabari and Bonazzola.

1. Lack of Priority Date Entitlement

As discussed in Ground 1B, "the processor" of the "computer device" recited in [9.1] is configured to perform the operative steps of [9.6]-[9.8]. This configuration was absent from original claim 9 (filed 7/1/2016; MS-1002, 37-39) and absent from the specification (MS-1015, 2-36, *see also* MS-1014), and was introduced into the record for the first time via an amendment (shown below) submitted 10/2/2017. *Id.*, 131. MS-1003, ¶¶64-78, 333-336.

9. An apparatus that performs automatic management of a virtualization environment in a computer network comprising:

a virtualization environment that has an inventory that includes network resources, virtual assets and a virtualization layer that manages the virtual assets;

a virtual mapping layer that determines an inventory of network resources and virtual assets available to the virtualization environment, wherein the virtual mapping layer determines the inventory without regard to processors, operating systems, virtualization platforms, and application software of the virtualization environment; and

a provisioning manager executed by a processor that communicates with the virtual mapping layer, the processor being configured to receive a virtual asset provisioning request that specifies parameters to be considered for the virtual asset provisioning request, automatically provision a particular virtual asset to a particular physical resource in the virtualization system that matches the parameters in the virtual asset provisioning request, assign the provisioned virtual asset to the virtual asset provisioning request and provide user access to the provisioned virtual asset.

MS-1002, 38.

9. (Currently Amended) An apparatus that performs automatic management of a virtualization environment in a computer network comprising:

a virtualization environment having a computer device with a processor and memory and a control center computer coupled to the computer device, the virtualization environment having ~~that has~~ an inventory that includes network resources and [[,]] virtual assets being software instantiations of computer systems hosted on the computer device and a virtualization layer executed by the computer device that manages the virtual assets;

the control center computer executing a virtual mapping layer that determines an inventory of network resources and virtual assets available to the virtualization environment, wherein the virtual mapping layer determines the inventory without regard to processors, operating systems, virtualization platforms, and application software of the virtualization environment; and

the control center computer executing a provisioning manager ~~executed by a processor~~ that communicates with the virtual mapping layer, the processor being configured to receive a virtual asset provisioning request that specifies parameters to be considered for the virtual asset provisioning request, automatically provision a particular virtual asset to a particular physical resource in the virtualization system that matches the parameters in the virtual asset provisioning request, assign the provisioned virtual asset to the virtual asset provisioning request and provide user access to the provisioned virtual asset.

MS-1002, 131.

Critically, the specification does not describe or suggest the host machine's processor performing the provisioning operations recited in [9.6]-[9.8]. *Id.* Rather, when addressing those operations, which it does, the specification calls out the central computer system processor and does not indicate whether they are performed also by the processor of the host computer coupled thereto. MS-1003, ¶77. This is not to say that the specification forecloses the possibility of them being performed by the host computer processor, together with the central computer processor, or independently. And indeed, this ground establishes that a POSITA would indeed find it obvious to perform those operations on the host computer processor based on those and other disclosures. *Id.*, ¶¶68-72. But "[e]ntitlement to a filing date extends only to the subject matter that is disclosed; not to that which is obvious...." *Research Corp. Techs., Inc. v. Microsoft Corp.*, 627 F.3d 859, 870 (Fed. Cir. 2010) (internal citations omitted). A proper priority claim to an earlier-filed application requires its disclosure to sufficiently convey "to a person of skill in the art that the patentee had possession of the claimed invention at the time of the application, i.e., that the patentee invented what is claimed." *Lizardtech, Inc. v. Earth Res. Mapping Inc.*, 424 F.3d 1336, 1345 (Fed. Cir. 2005); *Purdue Pharma L.P. v. Faulding Inc.*, 230 F.3d 1320, 1323 (Fed. Cir. 2000).

For reasons noted, the originally-filed application did not sufficiently convey

to a POSITA that patentee had possession of the "processor" configuration ultimately recited in claim 9, leaving the claim without a proper claim to priority. Accordingly, Pabari (MS-1012) and Bonazzola (MS-1013) are each available as prior art, and in combination, render claims 9-16 obvious.

2. **Pabari-Bonazzola**

(a) Pabari

Pabari is a patent application publication associated with US Patent No. 9,444,762, which is the parent of the '050 patent (by virtue of continuation), and thus shares the same disclosure as the '050 patent. *See* §II.

(b) Bonazzola

In Bonazzola, a setup component 130 executes on a host machine's processor to configure the host machine and locally provision a virtual machine without relying on a separate, dedicated management server. MS-1013, [0011]-[0013], FIGS. 1-3. A user provides storage, network, VM, and management system configuration information to a setup component 130 which then uses the information to set up network connectivity, prepare storage, create and start a VM, install an operating system. *Id.*, [0022]-[0026], [0030]-[0033]; MS-1003, ¶¶103-104.

(c) Combination

A POSITA would have combined Pabari and Bonazzola for implementation

details on how to receive and process provisioning requests at the processor of a computer device coupled to a central computer system and its processor (e.g., CPU 306 of computer device 300). MS-1003, ¶¶136-138. Bonazzola's setup component 130 (executing on a host machine's processor) locally performs provisioning operations (e.g., receiving user-specified configuration parameters, configuring storage and networking, creating and deploying a VM on the host machine, and registering the provisioned VM with a management system). MS-1013, [0011], [0021]-[0026], FIGS. 2-3. A POSITA would have understood these provisioning techniques to be applicable to provisioning of virtual assets generally, because Bonazzola's management system 140 "provision[s] new VMs" and "creat[es] virtual machines" using "VM templates that are used for creating new VMs with predefined settings and preinstalled disk images." *Id.*, [0020].

A POSITA would have recognized that Bonazzola's teachings would have improved Pabari's virtualization management architecture by enabling a processor of a host device (e.g., CPU 306 of computer device 300) to execute provisioning operations. MS-1003, ¶¶139-140. Bonazzola's teachings "improve[s] virtualization systems by allowing the management system to be deployed using less hardware than previous implementations (as the management system is not utilizing a separate machine to execute)." MS-1013, [0013]. This configuration would have also reduced reliance on the Control Center for provisioning

workflows thereby avoiding system architecture with a single point of failure, improved provisioning latency by enabling local execution on the host, and enabled provisioning in environments where connectivity to the Control Center is intermittent or unavailable. MS-1003, ¶¶139. Thus, by local execution of the operations recited in [9.6], [9.7], and [9.8] by a host device, Pabari-Bonazzola would have advanced Pabari's overall objective of providing automated management of virtualization environments across computer networks. *Id.*; *see* MS-1012, [0018].

Such a combination would have involved applying a known host-level provisioning technique (Bonazzola's setup component executing on the host machine's processor) to a known virtualization management architecture (Pabari's Control Center and host device arrangement) to yield predictable results. *KSR*, 550 U.S., at 415-421. A POSITA would have also reasonably expected to succeed in combining the teachings of Pabari and Bonazzola as the combination would have required only routine knowledge because the addition of Bonazzola's host-local provisioning component to Pabari's computer device would have involved modifications of software (specifically, to execute a local provisioning component similar to Bonazzola's setup component 130) in a manner that was well within the POSITA's skill. MS-1003, ¶¶141-142.

3. Analysis

(a) Claim 9

[9.pre], [9.1]-[9.5]

Pabari-Bonazzola renders these claims obvious based on the disclosures of Pabari identified below in combination with Bonazzola. MS-1003, ¶¶337-338.

Limitation	**Pabari**
[9.pre]	MS-1012, [0018], [0045]
[9.1]	*Id*., [0045], [0046], [0064], FIGS. 1-3
[9.2]	*Id*., [0064], FIG. 3
[9.3]	*Id*., [0049], FIG. 2
[9.4]	*Id*., [0045]; *see also* [0018]
[9.5]	*Id*., [0049], [0045], FIG. 2

[9.6]

In Pabari-Bonazzola, a host machine's (***computer device***) processing device (e.g., 402) (***processor***) would have executed a setup component 130 that receives configuration information (***virtual asset provisioning request***) from a user specifying parameters for provisioning a VM locally on the host, including storage configuration (e.g., "shared storage connection path and a storage domain and datacenter name"), network configuration (e.g., "Network Interface Card (NIC) information"), VM configuration (e.g., "memory size, number of virtual CPUs for

the VM"), and management system configuration (e.g., "host name and password"). MS-1013, [0022]-[0023], [0030], FIG. 2 (steps 205-250). The setup component 130 generates installation settings based on these user-specified inputs (***parameters to be considered for the virtual asset provisioning request***). *Id.*, [0030]; MS-1003, ¶¶339-342.

[9.7]

Pabari-Bonazzola's processing device (e.g., 402) (***processor***) in the host machine (***computer device***) would have executed the setup component 130 to automatically provision a VM (***particular virtual asset***) to the host machine's physical resources (***particular physical resource***) matching the user-specified parameters (***parameters in the virtual asset provisioning request***). MS-1013, [0024], [0032]-[0033], FIG. 3 (steps 305-330). The setup component 130 performs these steps "automatically and without user involvement." *Id.*, [0024]; *see also id.*, [0032]-[0033]; MS-1003, ¶¶343-346.

[9.8]

Pabari-Bonazzola's processing device (e.g., 402) (***processor***) would have executed the setup component 130 to assign the provisioned VM (***provisioned virtual asset***) to the provisioning workflow (***virtual asset provisioning request***) and provide user access to the provisioned VM (***provisioned virtual asset***). MS-1013, [0020], [0025]-[0026], FIG. 3 (steps 335-355). As to assigning, the setup

component 130 would have sent a command to the management system to add the host machine to a list of hosts associated with the management system (step 350) and wait for confirmation before restarting the VM (step 355). *Id.*, [0026]. As to providing user access, the setup component 130 provides "the user a temporary password to connect to the VM 115" during provisioning, and once operational, the management system manages "connection protocols between clients 101 and VMs." *Id.*, [0025]; *see id.*, [0020]; MS-1003, ¶¶347-351.

(b) Claims 10-16

Pabari-Bonazzola renders these claims obvious based on the disclosures of Pabari identified below in combination with Bonazzola. MS-1003, ¶352.

Claim	**Pabari**
10	MS-1012, [0061], [0176], FIGS. 2, 18
11	*Id.*, [0055]
12	*Id.*, [0055]
13	*Id.*, [0061], [0176], FIG. 18
14	*Id.*, [0048], [0170], FIG. 14
15	*Id.*, [0158], [0168], [0169], FIGS. 12, 13
16	*Id.*, [0166], [0176], FIG. 11

IV. CONCLUSION

The Challenged Claims are unpatentable. Microsoft 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 ’050 patent is the subject of civil action *ToutVirtual, Inc. v. Microsoft Corporation*, 2-25-cv-00943 (EDTX) filed September 8, 2025. The ’050 patent is subject to a terminal disclaimer with respect to U.S. Pat. No. 9,444,762. MS-1002, 199, 255-56. Microsoft is not aware of any disclaimers, reexamination certificates or petitions for *inter partes* review for the ’050 patent. Microsoft is not aware of any civil actions concerning the ’050 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: IPR42342-0155IP1@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 IPR42342-0155IP1@fr.com

D. Service Information

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

Petitioner consents to electronic service by email at IPR42342-0155IP1@fr.com.

Respectfully submitted,

Dated: March 10, 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. IPR2026-00296)

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 13,981 words, which is less than the 14,000 allowed under 37 CFR § 42.24.

Dated: March 10, 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 10, 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