

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,
Petitioner,

v.

EDGE NETWORKING SYSTEMS, LLC,
Patent Owner.

IPR2025-00617
Patent 10,893,095 B1

Before NABEEL U. KHAN, STACY B. MARGOLIES, and
JASON M. REPKO, *Administrative Patent Judges*.

REPKO, *Administrative Patent Judge*.

DECISION
Denying Institution of *Inter Partes* Review
35 U.S.C. § 314

I. INTRODUCTION

Microsoft Corporation (“Petitioner”) filed a petition (Paper 2, “Pet.”) requesting *inter partes* review of claims 1, 2, 6–8, 10–13, and 15 of U.S. Patent No. 10,893,095 B1 (Ex. 1001, “the challenged patent”). Edge Networking Systems, LLC (“Patent Owner”) filed a Preliminary Response. Paper 9 (“Prelim. Resp.”).

To institute an *inter partes* review, we must determine “that there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition.” 35 U.S.C. § 314(a). For the reasons discussed below, Petitioner has not shown a reasonable likelihood that it would prevail with respect to any claim challenged in the Petition. Thus, we do not institute an *inter partes* review.

A. *Related Matters*

According to the parties, the challenged patent is involved in *Edge Networking Systems, LLC v. Microsoft Corporation*, Case No. 1:24-cv-00215-DAE (W.D. Tex.); and *Edge Networking Systems LLC v. Amazon.com, Inc.*, Case No. 2-24-cv-00887 (E.D. Tex.). Pet. 82; Paper 3, 2.

B. *The Challenged Patent*

The challenged patent relates to a Distributed Software Defined Network (dSDN). Ex. 1001, Abstract. A purpose of the disclosed dSDN is to harmonize application deployment across different hardware independent of the vendor. *Id.* According to the challenged patent, this simplifies the entire deployment lifecycle. *Id.* The challenged patent uses what it calls an “fxDevice.” *Id.* at 10:18–21. The fxDevice is a network device “that is powered by a sandboxing operating system named flexible operating system

(fxOS) 302a.” *Id.* at 10:13–15. The fxOS can extend several aspects of an existing OS with network and routing layers and support for operating as the wireless infrastructure. *Id.* at 10:18–21.

C. Claims

Of those challenged, claims 1 and 15 are independent. Claim 1 is reproduced below.

1. A system for processing data packets in a network, the system comprising:

a programmable network device hosting a plurality of first network applications;

a programmable cloud device hosting a plurality of second network applications;

wherein at least one of the plurality of first network applications in the programmable network device and at least one of the plurality of second network applications in the programmable cloud device are in secure communication with each other to form a distributed application;

the programmable network device and programmable cloud device are each powered by a sandboxing operating system which facilitates deployment of the plurality of first and second network applications and facilitates upgrades of the first and second network applications with substantially no interruption to operation of the programmable network device and programmable cloud device;

wherein the programmable network device verifies authenticity of the upgrades to the plurality of first network applications and the programmable cloud device verifies authenticity of the upgrades to the plurality of second network applications and wherein the verification is based on unique security keys associated with each of the plurality of first and second network applications;

an application repository storing distributed applications which have been tested for installation in the programmable network device and programmable cloud device; and

an application management portal coupled to the programmable network device, programmable cloud device and the application repository, wherein the application management portal manages usage of the distributed applications on the programmable network device and programmable cloud device.

Ex. 1001, 37:41–38:42.

D. Evidence

Name	Reference	Exhibit No.
Vasell patent	US 6,496,575 B1, filed June 7, 1999	1004
Alves	Excerpts from A. de Castro Alves, <i>OSGi in Depth</i> , Manning Publications Company (2012)	1008
Hall	Excerpts from R. Hall, <i>OSGi in Action - Creating Modular Applications in Java</i> , Manning Publications Company (Apr. 2011)	1009
Rellermeyer	J. S. Rellermeyer and S. Bagchi, <i>Dependability as a Cloud Service – A Modular Approach</i> , IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN 2012) 1–6 (2012) ¹	1011
1st Vasell Provisional	U.S. Provisional Patent Application No. 60/088,437, filed on June 8, 1998, claimed as a priority application by the Vasell patent	1015
2 nd Vasell Provisional	U.S. Provisional Patent Application No. 60/123,971, filed on March 12, 1999, claimed as a priority application by the Vasell patent	1016

¹ Pet. iv. (Petitioner's Exhibit List).

E. Asserted Ground

Petitioner asserts that claims 1, 2, 6–8, 10–13, and 15 are unpatentable on the following ground. Pet. 23.

Claims Challenged	35 U.S.C. §²	Reference(s)/Basis
1, 2, 6–8, 10–13, 15	§ 103	Vasell, ³ Alves, Hall, Rellermeyer

II. ANALYSIS

A. Level of Ordinary Skill in the Art

According to the Petition,

[Persons of ordinary skill in the art (“POSITAs”)] as of June 2013 would have had a bachelor’s degree in computer science, computer engineering, or equivalent degree, and approximately three years of experience working in the computer science or engineering field. Additional experience might substitute for less education and vice versa. To that end, POSITAs in June 2013 would have been knowledgeable about the design and management of networked systems and virtualization technologies, and familiar with operating/distributed systems and security and privacy techniques.

Pet. 14 (citing Ex. 1003 ¶ 59).⁴ Patent Owner does not contest this in its Preliminary Response. Prelim. Resp. 39. We apply Petitioner’s definition of the level of ordinary skill in the art here.

² Congress amended § 103 when it passed the Leahy-Smith America Invents Act (AIA). Pub. L. No. 112–29, § 3(c), 125 Stat. 284, 287 (2011). Petitioner asserts that the version amended by the AIA applies here. Pet. 23.

³ We use Petitioner’s nomenclature in which Petitioner collectively refers to the Vasell patent (Ex. 1004) and its corresponding provisional applications (Exs. 1015, 1016) as “Vasell.”

⁴ Erez Zadok, Ph.D. submitted a declaration on behalf of Petitioner. See Ex. 1003.

B. Obviousness

Petitioner asserts that claims 1, 2, 6–8, 10–13, and 15 are unpatentable as obvious over Vasell, Alves, Hall, and Rellermeyer. Pet. 23.

1. Open Services Gateway initiative (OSGi)

Vasell, Alves, Hall, and Rellermeyer all generally relate to OSGi. *See infra* Sections II.B.2–3. OSGi allows Java code to be modularized and managed as “services.” Ex. 1003 ¶ 193. OSGi uses an execution environment based on the Java Runtime Environment (JRE), including the Java virtual machine (JVM). *Id.* OSGi dynamically installs and uninstalls Java modules, referred to as “bundles.” *Id.* Administrators can search for, download, and install OSGi bundles from a “bundle repository.” Ex. 1003 ¶ 203.

2. Vasell

Vasell describes a distributed-system architecture that Petitioner asserts “was used as the foundational architecture for the first OSGi specification in January 2000.” *See* Ex. 1003 ¶ 179; Pet. 18. Vasell’s Figure 2, below, shows a service gateway system for connectivity-based services. Ex. 1004, 4:30–31.

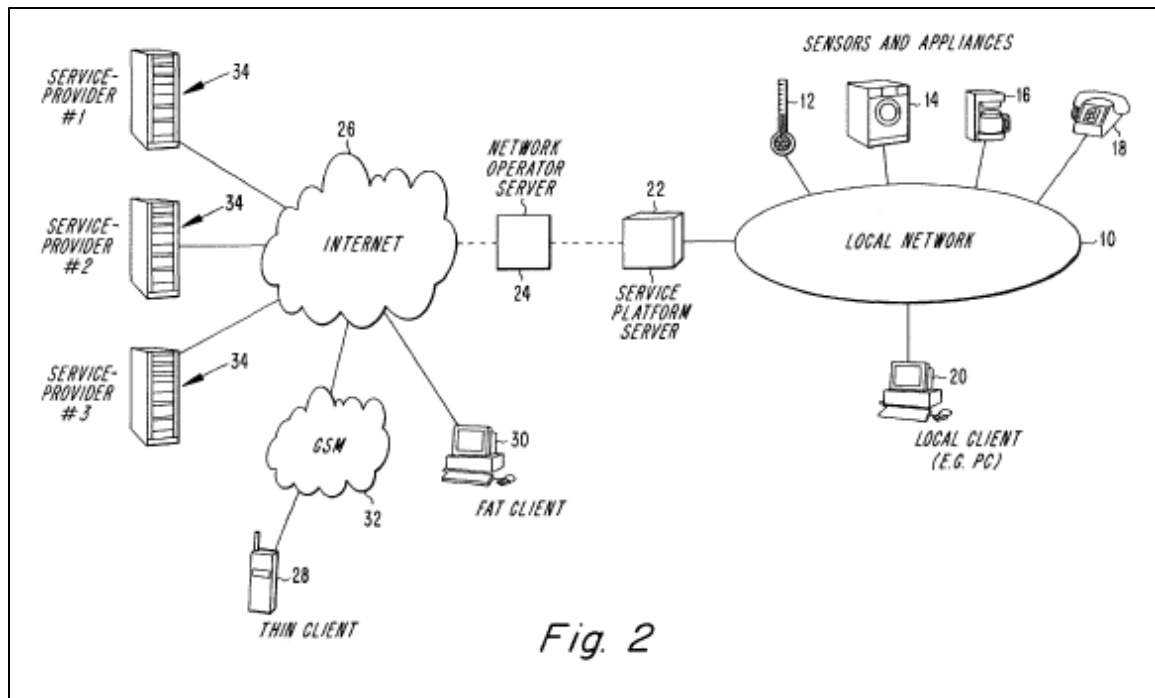


Figure 2 shows local client 20 and household sensors and appliances, including thermostat 12, washing/drying appliance 14, coffee maker 16, and telephone 18. *Id.* at 5:36–60. Service platform server 22 is installed at the user’s location and connected to the user’s local network 10. *Id.* at 5:64–6:4. Service platform server 22 interacts with the local devices to implement connectivity-based services. *Id.* at 6:5–8. For example, telephony services may interact with the user’s landline or wireless services for billing, special calling plans, voicemail, or Internet protocol (IP) telephony. *Id.* at 6:37–39.

Once the user has been confirmed as a subscriber, network operator server 24 maintains and controls service platform server 22. *Id.* at 8:16–20. Network operator server 24 can install software from service provider equipment 34 to server 22 to enable the user to control the devices in the user’s home. *See id.* at 8:16–46, 59–64. Terminals (thin client 28 and fat client 30) allow the user to interact with the services remotely. *Id.* at 7:21–

37. For example, the user can operate client 28 or 30 to monitor thermostat 12 or change its temperature settings over GSM 32. *Id.* at 7:21–36.

The distributed applications are called “service applications.” *Id.* at 12:11–29. Vasell’s Figure 6 shows an example implementation of a connectivity-based service. *Id.*

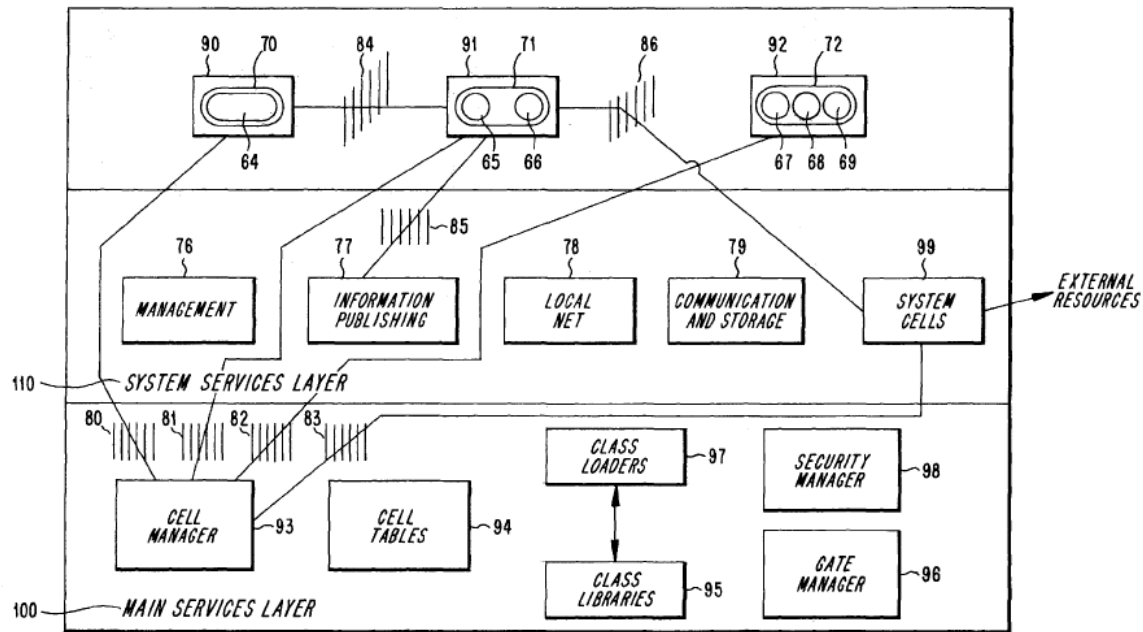


FIG. 6

Figure 6 shows three service applications (70–72) in three cells (90–92) and seven gates (80–86) connecting various system elements across system services layer 110 and main services layer 100. *Id.* at 12:21–23. System services layer 110 includes management system service 76, information publishing system service 77 for publishing or transmitting information for display, and local net system service 78, among other things. *Id.* at 22:32–53. Service applications 70–72 include boxlets 64–69. *Id.* at 13:28–29. Boxlets 64–69 have software code, programs, subroutines, arguments, or programming logic. *Id.* at 13:29–33. Cells 90–92 represent available resources. *Id.* at 13:50–51. Cells 90–92 are implemented using

class loaders 97 and provide encapsulation, protection, and access restrictions for service applications 70–72. *Id.* at 13:64–67. Cells are disabled by cell manager 93. *Id.* at 16:55–57. Cells remain disabled until loaded in cell table 94 or re-enabled by cell manager 93 or by the system management application program interface (API). *Id.* at 16:56–60.

Cells are associated with main libraries containing accessible classes (class libraries 95). *Id.* at 14:17–20. For example, a family may subscribe to a financial services application that provides checking-account privileges to all family members but restricts stock-trading privileges to only the adults in the family. *Id.* at 14:37–44. To enable these permissions, the financial services application could be arranged in two cells. *Id.*

Trusted, privileged system cells 99 have the authority to access external resources, such as devices, files, sockets, or programs. *Id.* at 17:40–50. Cells 90–92 communicate with each other via gates 80–86. *Id.* at 17:40–45. For example, Figure 6 shows system cells 99 in communication with cell 91 via gate 86. Gates 80–86 work locally within a single local operating environment, such as a JVM. *Id.* at 18:27–30. Remote gates may reside in multiple JVMs. *Id.* at 18:29–30.

3. *Alves, Hall, and Rellermeyer*

According to Petitioner, Alves, Hall, and Rellermeyer describe technical details for implementing Vasell’s teachings. Pet. 36. Alves is an excerpt from a textbook, *OSGi In Depth*, describing, among other things, how to implement OSGi-compatible systems in a cloud environment. Ex. 1008, 264–66. Similarly, Hall consists of excerpts from a textbook, *OSGi In Action*, describing how the bundles are deployed and how applications are secured. See Ex. 1009, 331–34 (describing deployment),

438–39 (describing Java’s security architecture and signing bundles). Rellermeyer describes turning OSGi services into elastic deployment units replicated across multiple cloud nodes. Ex. 1011, 4.

4. *Claims 1 and 15*

Claims 1 and 15 are the only independent claims challenged. Claim 1 recites, in part,

[1.4] the programmable network device and programmable cloud device are *each powered by a sandboxing operating system* which facilitates deployment of the plurality of first and second network applications and [1.5] facilitates upgrades of the first and second network applications with substantially no interruption to operation of the programmable network device and programmable cloud device.

Ex. 1001, 37:55–62 (emphasis and labels added). The Petition refers to these limitations as [1.4] and [1.5]. Pet. 55, 53. Claim 15 recites similar limitations as part of the claimed system. Ex. 1001, 40:17–24. All other challenged claims depend from claim 1.

Petitioner has not sufficiently explained how the combination of Vasell, Alves, Hall, and Rellermeyer teaches or suggests a programmable network device and a programmable cloud device that are each “powered by a sandboxing operating system,” as required by all challenged claims. Our reasoning follows.

a. *The Petition*

i. *Petitioner’s Claim Construction*

Petitioner (1) argues that each claim term in the challenged claims should be given its plain and ordinary meaning, (2) asserts that the challenged claims are unpatentable under either party’s construction in the

related district court case, and (3) for certain limitations, including limitations [1.4] and [1.5], “relies on [Patent Owner’s] identification . . . of ‘specific portions of the specification that describe the structure’ corresponding to the recited functions” to the extent that Patent Owner argues or the Board finds that the limitations should be interpreted as means-plus-function limitations under 35 U.S.C. § 112 (f). Pet. 10–11 (citing Ex. 1070).

For limitations [1.4] and [1.5], Petitioner provides string citations of parts of the specification with little additional explanation or analysis. *Id.* at 11–12. Petitioner quotes the challenged patent’s description of (1) how a sandboxing operating system (OS) allows applications to run and interact with the device’s resources and (2) how the OS ensures isolation to all the application instances to continue running during the upgrade without interrupting it. *Id.*

ii. *Petitioner’s Claim Analysis*

Petitioner argues that it would have been obvious to combine Alves, Hall, and Rellermeyer with Vasell because, among other reasons, those references describe technical details for implementing Vasell’s teachings, particularly those relating to OSGi. Pet. 36–38. As for the “sandboxing operating system” recited in limitation [1.4], Petitioner’s analysis relies solely on Vasell. *See id.* at 53–55. The “sandboxing operating system” is recited in limitation [15.2], and Petitioner refers to the analysis of limitation [1.4], stating “Vasell alone, or in view of Alves, discloses this limitation per [1.4].” *Id.* at 77. There is no additional analysis of how Alves teaches or suggests the “sandboxing operating system” for either limitation [1.4] or

limitation [15.2], so we understand Petitioner to rely solely on Vasell for the “sandboxing operating system” in [15.2]. *See id.*

Specifically, as for [1.4], Petitioner argues that Vasell discloses that service application components are hosted on a JVM running on an OS:

Per [1.0]-[1.3], Vasell discloses the distributed “service applications” as being “modularized”, with each distributed service application component respectively hosted on *a JVM running on an OS* on either a service platform server or remote (cloud) server, and where each *OS hosting each JVM* was a “UNIX-like standard [OS]” provided it has “fairly strong resource control and protection mechanisms to ensure application integrity” such that “errors in software developed by different service providers” cannot result in the “applications... interfer[ing] with one another.”

Id. at 53–54 (colors removed, italics added, alteration in original). Here, Petitioner refers to both the JVM and the “OS hosting each JVM.” *Id.*

Notably, Petitioner relies upon the resource control and protection mechanisms of the “OS hosting each JVM”—not the JVM itself. *Id.*

Petitioner’s declarant, Dr. Zadok, testifies that Vasell’s way of powering the service gateway system server and facilitating software deployment is substantially similar to the description of powering and facilitating in the challenged patent. Ex. 1003 ¶ 472.

b. Patent Owner’s Preliminary Response

Patent Owner argues that Petitioner does not explain how Vasell teaches the claimed sandboxing OS—specifically, that the programmable network device and programmable cloud device are each “powered by a sandboxing operating system.” Prelim. Resp. 45–47. According to Patent Owner, this limitation requires providing sandboxing at the OS level. *Id.* at 45. Patent Owner argues that Petitioner repeatedly describes Vasell’s JVM

as hosted by or running on an OS and thus distinct from the OS itself. *Id.* at 46 (citing Pet. 43, 53). Patent Owner argues that Petitioner does not explain how a JVM can be an OS if the JVM itself runs on the OS. *Id.* at 41–42. If Petitioner intends to map Vasell’s OS to the claimed OS, Patent Owner argues that there is no explanation of how Vasell’s cell manager 93, relied upon for the upgrade feature (Pet. 56), is part of the OS. Prelim. Resp. 49–50. Patent Owner argues that Petitioner does not explain how Vasell discloses that an OS facilitates the upgrades, as required by the claim. *Id.*

c. Analysis: Insufficient Explanation of How Vasell Teaches or Suggests the Claimed Sandboxing OS

As explained below, various statements in the Petition suggest that Petitioner regards (1) Vasell’s JVM as the sandboxing OS, (2) Vasell’s UNIX-like OS as the sandboxing OS, or (3) the JVM and the UNIX-like OS together as the sandboxing OS. The Petition is thus deficient because Petitioner has not clearly identified how Vasell teaches the claim requirement that the programmable network device and the programmable cloud device are each “powered by a sandboxing operating system.”

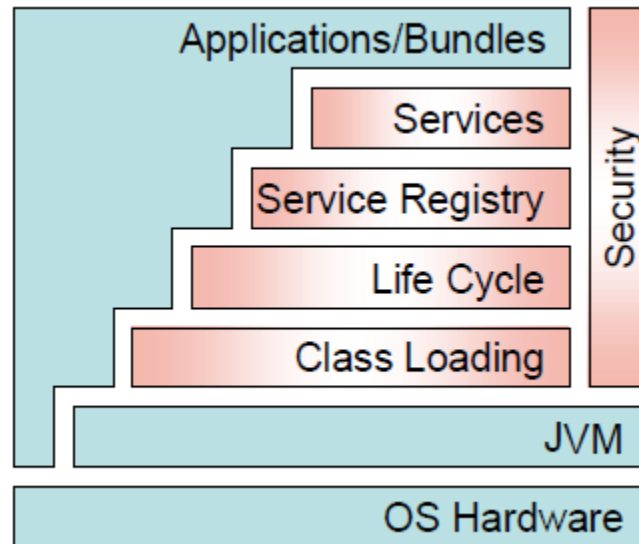
i. JVM or the UNIX-like OS as the Sandboxing OS

Petitioner does not sufficiently explain how Vasell teaches or suggests the disputed limitations if the JVM or the UNIX-like OS were considered as individually providing the features of the sandboxing OS. Pet. 10–11.

(a) Sandboxing in Operating Systems versus Sandboxing using a JVM

Petitioner and Dr. Zadok recognize that a JVM and the OS hosting the JVM are distinct and provide different ways of sandboxing. See Pet. 14–16

(“State of the Art”); Ex. 1003 ¶¶ 117–135. For example, Dr. Zadok cites the Figure below, which shows an OS hosting a JVM in an OSGi Service Platform. Ex. 1013, 11, *cited in* Ex. 1003 ¶ 194.



The figure above shows an OS layer and a layer for the JVM. Ex. 1013, 11. According to Dr. Zadok, sandboxing uses a layer logically below the isolated entities to enforce isolation. Ex. 1003 ¶ 116. Examples of isolation include an OS “isolating processes and containers from each other (and the OS itself)” and the JVM “isolating Java programs from each other.” *Id.*

As for sandboxing at the OS layer, according to Dr. Zadok, there were various ways to ensure that only processes with proper authorization from the OS can operate on files, memory segments, the CPU, and other system resources. *Id.* ¶ 117. For example, FreeBSD used a “jail” to limit a process’s interaction with other processes outside the jail. *Id.* ¶ 129.

According to Dr. Zadok, the JVM layer also provided a sandbox. *Id.* ¶ 116. According to a source quoted by Dr. Zadok, Java programs could not directly manipulate a computer’s hardware or make direct calls to the OS. *Id.* ¶ 133 (quoting Ex. 1045, 44–45). Rather, the programs run on the JVM.

Id. According to the same source, the Java execution environment was like “a place where a child can build things and break things and generally not get hurt and not hurt the outside world.” *Id.*

In the sub-section “Sandboxing Operating Systems” under the section titled “State of the Art,” cited in Petitioner’s analysis of limitation [1.4] (Pet. 54 (citing State of the Art § V.F.1)), Petitioner refers to the features found in traditional OSs:

Traditional computer OSs included mechanisms to “sandbox” software programs (*e.g.*, confining/restricting what resources an application accesses and when).

Id. at 14–15. Later in that section, Petitioner also refers to sandboxing in Java:

Java’s security model provides a well-known *example of a sandboxed OS*, in that the runtime environment, typically a “Java Virtual Machine (JVM)”, resided on the host computer’s OS and confined all Java applications to an isolated area—the “sandbox.” Specifically, the JVM’s well-known “sandbox” provided a restricted environment for running untrusted code (restricting access to resources inside the sandbox) alongside trusted code (full access to system resources).

Id. at 15–16 (emphasis added).

(b) *Petitioner’s Reliance on Features from the OS and the JVM*

In the analysis of limitation [1.4], Petitioner relies on the features of the UNIX-like OS that provides resource control and protection mechanisms. Pet. 53–54. Specifically, Petitioner asserts that “each OS hosting each JVM was a ‘*UNIX-like standard [OS]*’ provided it has ‘fairly strong-resource control and protection mechanisms to ensure application integrity’ such that ‘errors in software developed by different service providers’ cannot result in the ‘applications... interfer[ing] with one

another.” *Id.* (citing Ex. 1016, 4, 6–9, Figs. 2, 5; Ex. 1004, 3:18–23, 18:27–30). That is, Petitioner is referring to the mechanisms of the UNIX-like standard OS, not the JVM, for providing sandboxing. *Id.* The part of Vasell that Petitioner quotes here states, in full,

Any UNIX-like standard system, including Open Source, can be chosen as an operating system. The system must have fairly strong resource-control and protection mechanisms to ensure application integrity.

Ex. 1016, 7. This paragraph in Vasell follows a list of what comprises the system software, which has separate bullets for “an operating system” and “a Java runtime in the form of a Java virtual machine (JVM).” *Id.* Thus, Petitioner’s analysis specifically relies upon the features of the UNIX-like OS in the analysis of the claimed sandboxing OS. Pet. 53–54.

In the analysis of limitation [1.5], on the other hand, Petitioner relies on the features of the JVM by identifying Vasell’s cell manager as facilitating uninterrupted “upgrades.” *Id.* at 56. We agree with Patent Owner’s argument with respect to limitation [1.5] that Petitioner neither explains how Vasell’s cell manager 93 is part of the UNIX-like standard OS or how the UNIX-like standard OS facilitates the recited upgrades. Prelim. Resp. 50. Rather, cell manager 93 uses the features of the JVM: cell manager 93 provides one of the main gates 80–82 to each one of the cells 90–92 at creation (Ex. 1004, 20:11–12), the gates represent inter-cell connections (*id.* at 18:23–25), and those gates work locally within the JVM (*id.* at 18:27–29). Petitioner has not explained how the JVM can be the OS if the JVM runs on the OS. Prelim. Resp. 41–42.

Thus, Petitioner does not clearly show how Vasell teaches the claimed sandboxing OS based on the features of (1) the JVM alone or (2) the UNIX-like OS alone as corresponding to the recited sandboxing OS.

ii. *JVM and the UNIX-like OS as the Sandboxing OS*

Petitioner also does not sufficiently explain how Vassel teaches or suggests the disputed limitations if both the JVM and the UNIX-like OS were considered together as the sandboxing OS under either party's construction in the related litigation or the plain meaning of the term. Pet. 10–11.

(a) *Means-Plus-Function Interpretation*

Interpreting the claim as reciting a means-plus-function limitation, Petitioner does not identify a JVM hosted on an OS as the corresponding structure to carry out the functions recited in [1.4]. See Pet. 54–55. Instead, Petitioner argues that “Vasell’s structure to carry out the [1.4]-recited functions is the PO-indicated structure in the [challenged patent] or equivalent thereof, which states the sandboxing OS may be ‘built based on an existing sandboxed operating system.’” *Id.* Here, Petitioner provides no explanation of what it means to be “based on” an existing sandboxed OS. *Id.* For example, Petitioner never fully explains what the existing OS would be if a JVM running on an OS is itself an OS that is based on an existing OS.

Even so, Petitioner quotes only part of the paragraph from the challenged patent, omitting an important discussion about how the operating system can be built “based on” the OS:

The fxDevice 302 is a network device that is powered by a sandboxing operating system named flexible operating system (fxOS) 302a . . . The fxOS 302a may be built based on an existing sandboxed operating system (OS) (e.g. , Android®) *by extending several aspects of such an OS with routing/networking layers and supports for operating as the wireless infrastructure.*

Ex. 1001, 10:13–21. That is, the cited part of the challenged patent states that the sandbox-powered OS may be built by extending an existing OS. *Id.* By contrast, Petitioner does not explain how a JVM “hosted on” a server OS is “built based on” any existing OS. Consistent with Dr. Zadok’s testimony (Ex. 1003 ¶¶ 133–134), we understand the phrase “JVMs hosted on server OSs” to mean that a JVM runs on the existing OS, not that another OS is built from an existing OS (Ex. 1001, 10:14–24).

Thus, Petitioner does not show that Vasell teaches or suggests the disputed limitations if both the JVM and the UNIX-like OS were considered together as the sandboxing OS under the means-plus-function interpretation.

(b) Petitioner’s Claim Construction in the Related Litigation

Petitioner also does not show that Vasell teaches the disputed limitations if both the JVM and the UNIX-like OS were considered together as the sandboxing OS under Petitioner’s proposed construction in the Claim Construction Brief filed in the related litigation. *See* Ex. 1072, 7–8. In the brief, Petitioner argues that the claimed OS exists “alongside” an existing OS (*id.* at 8), not built based on it, as in the analysis of [1.4] (Pet. 54–55). Also, in the brief, Petitioner argues that an OS must have a kernel that can access the hardware. Ex. 1072, 7–8.

But the Petition does not identify the two OSs with kernels existing alongside each other. *See id.* In fact, as Patent Owner points out, the Petition lacks any analysis of kernels. Prelim. Resp. 33–34. So the Petition never fully explains how the JVM-UNIX-like OS combination meets Petitioner’s construction from the related litigation, despite arguing that “the Grounds render the Challenged Claims unpatentable under *either party’s*

construction” from the related litigation. Pet. 11 (citing Ex. 1070; Ex. 1072) (emphasis added).

(c) *Plain Meaning*

Although Petitioner states that the claims are unpatentable under the plain meaning of the terms, Petitioner does not explicitly state that the plain meaning of the term “operating system” is a JVM together with an operating system. See Pet. 10–11. As Patent Owner notes, “The Petition repeatedly describes Vasell’s JVM as being ‘hosted by’ or ‘running on’ an operating system, and thus distinct from the operating system itself.” Prelim. Resp. 43 (emphasis removed). We agree. Specifically, as discussed in Section II.B.4.c.i above, Petitioner discusses how the JVM and the UNIX-like system each provide sandboxing in different ways. Pet. 14–16. This is the same way that Vasell discusses OSs and JVMs, listing what comprises the system software and showing separate bullets for “an operating system” and “a Java runtime in the form of a Java virtual machine (JVM).” Ex. 1016, 7. To the extent Petitioner is asserting that the plain meaning of OS encompasses the JVM and the Unix-based OS, Petitioner’s showing is deficient for the reasons explained above. See Prelim. Resp. 41–42, 50; see Section II.B.4.c.i *supra*.

d. *Summary*

Petitioner has not sufficiently explained how the proposed combination teaches or suggests the sandboxing operating system of claim 1 and the corresponding limitation in claim 15. See Pet. 53–57, 77.

5. *Dependent Claims*

For the same reasons, Petitioner has not shown that the proposed combination teaches or suggests the subject matter recited in the challenged claims that depend from claims 1 and 15, as the additional analysis in the challenges to those claims do not remedy the deficiency described above. *See* Pet. 64–74.

III. CONCLUSION

Petitioner has not shown a reasonable likelihood of prevailing on the obviousness grounds for any claims. Thus, we do not institute an *inter partes* review. *See* 35 U.S.C. § 314(a).

IV. ORDER

It is ORDERED that the Petition is *denied*.

IPR2025-00617
Patent 10,893,095 B1

FOR PETITIONER:

Christopher Tyson
John Baird
Glenn Richeson
Patrick McPherson
DUANE MORRIS LLP
cjtyson@duanemorris.com
jmbaird@duanemorris.com
gdricheson@duanemorris.com
pdmcperson@duanemorris.com

FOR PATENT OWNER:

Brandon Theiss
Daniel Golub
VOLPE KOENIG
btheiss@vklaw.com
dgolub@vklaw.com

David Hecht
HECHT PARTNERS LLP
dhecht@hechtpartners.com