

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION
Petitioner,

v.

EDGE NETWORKING SYSTEMS, LLC,
Patent Owner.

U.S. Patent No. 11,695,823
Issue Date: July 4, 2023
Title: DISTRIBUTED SOFTWARE DEFINED NETWORKING

Inter Partes Review No.: IPR2025-00618

**PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO.
11,695,823 UNDER 35 U.S.C. §§ 311-319 and 37 C.F.R. §§ 42.1-.80, 42.100-
.107**

Mail Stop “PATENT BOARD”
Patent Trial and Appeal Board
U.S. Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

| | Page |
|--|-------------|
| I. INTRODUCTION | 1 |
| II. OVERVIEW | 1 |
| III. GROUNDS FOR STANDING (37 C.F.R. §42.104(A)) | 6 |
| IV. RELIEF REQUESTED (37 C.F.R. §42.22(a))..... | 6 |
| V. REASONS FOR THE REQUESTED RELIEF | 6 |
| A. '823 Patent Summary | 6 |
| B. Prosecution History | 10 |
| C. Claim Construction | 10 |
| D. Priority Date of the Challenged Claims | 11 |
| E. Person of Ordinary Skill in the Art | 12 |
| F. State of the Art | 12 |
| 1. Sandboxing Operating Systems | 12 |
| 2. Security for Software Downloaded/Implemented over the Internet | 14 |
| 3. Distributed Systems | 15 |
| VI. IDENTIFICATION OF CHALLENGES..... | 22 |
| A. Challenged Claims | 22 |
| B. Statutory Grounds for Challenges | 22 |
| VII. IDENTIFICATION OF HOW THE CHALLENGED CLAIMS ARE UNPATENTABLE | 23 |
| A. Prior Art in the Grounds..... | 23 |
| 1. Vasell..... | 23 |

| | | |
|-------|---|----|
| 5. | The Motivation to Combine..... | 36 |
| B. | Ground 1: Detailed Application of Vasell in view of Alves and Rellermeyer to Claims 1-2, 12-15 and 19 | 43 |
| | Claim 1 | 43 |
| | Claim 2 | 57 |
| | Claim 12..... | 58 |
| | Claim 13..... | 59 |
| | Claim 14..... | 60 |
| | Claim 15..... | 63 |
| | Claim 19..... | 63 |
| C. | Ground 2: Detailed Application of Vasell in view of Alves, Rellermeyer and Hall to Claims 3-5, 7-8 and 18 | 67 |
| | Claim 3 | 67 |
| | Claim 4..... | 68 |
| | Claim 5..... | 69 |
| | Claim 7..... | 71 |
| | Claim 8..... | 73 |
| | Claim 18..... | 73 |
| VIII. | Discretionary Denial Would be Inappropriate | 77 |
| | A. Discretionary Denial under the <i>Fintiv</i> Factors is Inappropriate..... | 77 |
| IX. | Mandatory Notices Under 37 C.F.R. §42.8..... | 80 |
| | A. Real Party-in-Interest (37 C.F.R. § 42.8(b)(1))..... | 80 |
| | B. Related Matters (37 C.F.R. § 42.8(b)(2))..... | 80 |
| | 1. Judicial Matters..... | 80 |

| | | |
|-----------|---|-----------|
| 2. | Related Patents..... | 81 |
| C. | Lead/Back-up Counsel (37 C.F.R. § 42.8(b)(3)): | 81 |
| D. | Notice of Service Information (37 C.F.R. § 42.8(b)(4)): | 82 |
| X. | CONCLUSION | 82 |

PETITIONER’S EXHIBIT LIST

| <i>Exhibit #</i> | <i>Description</i> |
|------------------|---|
| 1001 | U.S. Patent No. 11,695,823 (the “’823 Patent”). |
| 1002 | Prosecution history of the ’823 Patent (“’823 File History”). |
| 1003 | Declaration of Petitioner’s Expert Dr. Erez Zadok (“Zadok”). |
| 1004 | U.S. Patent No. 6,496,575 to Vasell (“Vasell Patent”). |
| 1005 | File history of the parent patent to the ’823 Patent, U.S. Patent No. 9,843,624 (the “’624 Patent”). |
| 1006 | File history of the patent from which the ’823 Patent was filed as a continuation application, U.S. Patent No. 10,893,095 (the “’095 Patent”). |
| 1007 | File history of the other patent from which the ’823 Patent claims priority in a chain of continuation applications, U.S. Patent No. 10,686,871 (the “’871 Patent”). |
| 1008 | Excerpts from A. de Castro Alves, “OSGi in Depth”, Manning Publications Company, 2012 (“Alves”). |
| 1009 | Excerpts from R. Hall, et al., “OSGi in Action - Creating Modular Applications in Java”, Manning Publications Company, April 2011 (“Hall”). |
| 1010 | J. E. Kim, et al., “Seamless Integration of Heterogeneous Devices and Access Control in Smart Homes,” 2012 Eighth International Conference on Intelligent Environments, Guanajuato, Mexico, June 26-29, 2012, pp. 206-213 (“Kim”). |
| 1011 | J. S. Rellermeyer and S. Bagchi, “Dependability as a cloud service - a modular approach,” IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN 2012), Boston, MA, USA, 2012, pp. 1-6 (“Rellermeyer”). |
| 1012 | S. Kächele, et al., “Component-based scalability for cloud applications”, Proceedings of the 3 rd International Workshop on Cloud Data and Platforms (CloudDP ‘13), April 14, 2013, Prague, Czech Republic, 19–24 (“Kächele”). |

| <i>Exhibit #</i> | <i>Description</i> |
|------------------|---|
| 1013 | OSGi Alliance, “About the OSGi Service Platform”, Technical Whitepaper, Revision 4.1, November 11, 2005, archived on November 30, 2005 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20051130032628/http://www.osgi.org/documents/collateral/TechnicalWhitePaper2005osgi-spooverview.pdf (“OSGi 2005 Whitepaper”). |
| 1014 | Declaration of June A. Munford. |
| 1015 | U.S. Provisional Patent Application No. 60/088,437, filed on June 8, 1998, claimed as a priority application in, and incorporated by reference into, EX1004 (“1 st Vasell Provisional”) (collectively, with EX1004 and EX1016, “Vasell”). |
| 1016 | U.S. Provisional Patent Application No. 60/123,971, filed on March 12, 1999, claimed as a priority application in, and incorporated by reference into, EX1004 (“2 nd Vasell Provisional”) (collectively, with EX1004 and EX1015, “Vasell”), which includes a copy of the article published as Idermark, T., Lilliestråle, M., & Vasell, J., “Ericsson’s e-box system—An electronic services enabler”, Ericsson Review, (1), 1999, 38-44, as confirmed by Ericsson’s website archived, for example, on March 3, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000303195310/http://www.ericsson.se/review/issues.taf and, for example, on February 16, 2003 and November 27, 2004 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20030216191220/http://www.ericsson.com/about/publications/review/1999_01/23.shtml and https://web.archive.org/web/20041127035720/http://www.ericsson.com/about/publications/review/1999_01/files/1999015.pdf respectively. |
| 1017 | “About Gatespace”, Gatespace AB website, archived on January 6, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000106145021/http://www.gatespace.com/ (“About Gatespace 2000”). |
| 1018 | “Carlstedt Research & Technology Forms New Company for Developing Software for ‘The Intelligent Home’ in Cooperation with Ericsson”, September 30, 1999, Gatespace AB website, archived on February 12, 2001 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20010212054449/http://www.gatespace.com/news/archive/19990930_en_v3.shtml (“Gatespace Formation Press Release”). |

| <i>Exhibit #</i> | <i>Description</i> |
|------------------|---|
| 1019 | “Ericsson e-services is a part of the Open Services Gateway Initiative (OSGI)”, Ericsson e-services website, archived on February 29, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000229111550/http://www.ericsson.com:80/wireless/products/ebox/links/osgi.shtml (“Ericsson E-Services WebPage”). |
| 1020 | “Fifteen industry leaders to create standard for bringing Internet-based services to the networked home”, March 1, 1999, Ericsson e-services website, archived on May 28, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000528223608/http://www.ericsson.com/wireless/products/ebox/news/990301.shtml (“OSGi Announcement on Ericsson E-Services WebPage”). |
| 1021 | “People”, Carlstedt Research & Technology AB website, archived on January 6, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000106204945/http://www.crt.se/about/people.en.html (“CR&T People 2000”). |
| 1022 | “Open Services Gateway Initiative: Charter”, OSGi website, archived on April 19, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000419181219/http://www.osgi.org/about/charter.html (“OSGi Charter 2000”). |
| 1023 | “Open Services Gateway Initiative: Officers”, OSGi website, archived on April 14, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000414133426/http://www.osgi.org/about/officers.html (“OSGi Officers 2000”). |
| 1024 | “Open Services Gateway Initiative Elects New Leadership”, Home Toys Inc. website, May 23, 2001, archived on July 8, 2001 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20010708123711/http://www.hometoys.com/releases/jun01/osgi01.htm (“OSGi Leadership 2001”). |
| 1025 | “Management Team”, Gatespace AB website, archived on February 8, 2002 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20020208031748/http://www.gatespace.com/company/management.shtml (“Gatespace Management 2002”). |
| 1026 | “About Us: Background”, Gatespace AB website, archived on February 6, 2002 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20020206205742/http://www.gatespace.com/company/ (“About Gatespace 2002”). |

| <i>Exhibit #</i> | <i>Description</i> |
|-------------------------|--|
| 1027 | “About Makewave: History in Making”, Makewave AB website, archived on August 16, 2007 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20070816163615/http://www.makewave.com/site.en/about/history.shtml (“About Makewave 2007”). |
| 1028 | The OSGi Alliance, “OSGi Service Platform Enterprise Specification”, Release 4, Version 4.2, March 2010 (“OSGi Enterprise Specification”). |
| 1029 | T. Forst, “Cisco Application eXtension Platform – OSGi Add-on as universal Middleware for Your Applications”, May 2008, AutomatedBuildings.com, archived on May 17, 2008 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20080517111602/www.automatedbuildings.com/news/may08/articles/prosyst/080427034829prosyst.htm (“Forst”). |
| 1030 | U.S. Patent Publication No. 2013/0086147 to Kashyap, filed October 3, 2011, published April 4, 2013 (“Kashyap”). |
| 1031 | Excerpts from A.S. Tanenbaum, “Modern Operating Systems”, Prentice Hall, 3 rd edition, 2008 (“Tanenbaum 2008”). |
| 1032 | Excerpts from A.S. Tanenbaum, “Computer Networks”, Prentice Hall, 3 rd edition, 1996 (“Tanenbaum 1996”). |
| 1033 | Excerpts from D. Downing, et al., “Dictionary of Computer and Internet Terms”, Barron’s Educational Series, Inc., 11 th edition, 2013 (“Downing”). |
| 1034 | Excerpts from E. Cole, et al., “Network Security Bible”, Wiley Publishing, Inc., 2 nd edition, 2009 (“Cole”). |
| 1035 | Curriculum Vitae (CV) of Petitioner’s Expert Dr. Erez Zadok. |
| 1036 | S. Soltesz, et al., “Container-based Operating System Virtualization: A Scalable, High Performance Alternative to Hypervisors”, In Proceedings of the 2 nd ACM SIGOPS/EuroSys European Conference on Computer Systems (EuroSys ‘07), March 21-23, 2007, Lisbon, Portugal, pp. 275-287, 2007 (“Soltesz”). |
| 1037 | <i>Intentionally Left Blank</i> |
| 1038 | <i>Intentionally Left Blank</i> |

| <i>Exhibit #</i> | <i>Description</i> |
|------------------|---|
| 1039 | Li, et al., “ExpoNet: A Flexible Platform for Concurrent Experiments on Programmable Infrastructure”, 2011 IEEE Global Telecommunications Conference - GLOBECOM 2011, Houston, TX, USA, 2011, pp. 1-5 (“Li”). |
| 1040 | <i>Intentionally Left Blank</i> |
| 1041 | P. Kriens, “The Bundle Repository”, OSGi Alliance Blog, April 7, 2006, archived on May 4, 2006 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20060504203641/http://www.osgi.org/blog/2006/04/bundle-repository.html (“Kriens OSGi Blog”). |
| 1042 | “Knoplerfish Pro Premium” and “Knoplerfish Pro Enterprise” Data Sheets, Makewave AB, 2010, respectively archived on February 20, 2011 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20110220123804/http://www.makewave.com:80/resources/docs/datasheets/MD-042-A1-knoplerfish_pro_premium_osgi.pdf and https://web.archive.org/web/20110220123819/http://www.makewave.com:80/resources/docs/datasheets/MD-047-A1-knoplerfish_pro_enterprise_osgi.pdf (“Knoplerfish”). |
| 1043 | R. Kawashima, “vNFC: A Virtual Networking Function Container for SDN-enabled Virtual Networks”, 2012 Second Symposium on Network Cloud Computing and Applications, London, UK, 2012, pp. 124-129 (“Kawashima”). |
| 1044 | Excerpts from A.S. Tanenbaum, “Modern Operating Systems”, Prentice Hall, 2 nd edition, 2001 (“Tanenbaum 2001”). |
| 1045 | Excerpts from S. Garfinkel, et al., “Web Security & Commerce”, O’Reilly & Associates, 1997 (“Garfinkel”). |
| 1046 | Oasis Standard Specification, “Web Services Security: SOAP Message Security 1.1 (WS-Security 2004), February 1, 2006, archived on February 6, 2007 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20070206143722/http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-soapmessagesecurity.pdf , and accessible from OASIS Web Services Security (WSS) Technical Committee web page, as archived on December 5, 2006 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20061205032600/http://oasis-open.org/committees/tc_home.php?wg_abbrev=wss#technical (“Oasis - SOAP Security”). |

| <i>Exhibit #</i> | <i>Description</i> |
|------------------|---|
| 1047 | <p>“Web Service Security Cheat Sheet”, The Open Web Application Security Project (OWASP), archived on June 12, 2012 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20120612205800/https://www.owasp.org/index.php/Web_Service_Security_Cheat_Sheet, and accessible from OWASP’s (the free and open software security community) web pages, as archived on June 10, 2012 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20120610001928/https://www.owasp.org/index.php/Main_Page, and as archived on June 10, 2012 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20120610091415/https://www.owasp.org/index.php/Cheat_Sheets (“OWASP - WSS Security”).</p> |
| 1048 | <p>“REST Security Cheat Sheet”, The Open Web Application Security Project (OWASP), archived on January 15, 2013 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20130115190935/https://www.owasp.org/index.php/REST_Security_Cheat_Sheet, and accessible from OWASP’s (the free and open software security community) web pages, as archived on January 4, 2013 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20130104230717/https://www.owasp.org/index.php/Main_Page, and as archived on January 15, 2013 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20130115215253/https://www.owasp.org/index.php/Cheat_Sheets (“OWASP – REST Security”).</p> |
| 1049 | <i>Intentionally Left Blank</i> |
| 1050 | <p>The Open Services Gateway Initiative, “Specification Overview”, Version 1.0, January 2000, archived on August 31, 2000 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20000831061202/http://www.osgi.org/about/specoverview.pdf (“OSGi 2000 Overview”).</p> |
| 1051 | <p>R. Hall, “Oscar Bundle Repository”, archived on June 30, 2004 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20040630163519/http://oscar-osgi.sourceforge.net:80/ (“OBR 2004”).</p> |
| 1052 | <p>Excerpts from B. Sosinsky, “Cloud Computing Bible”, Wiley Publishing, Inc., 2011 (“Sosinsky”).</p> |

| <i>Exhibit #</i> | <i>Description</i> |
|------------------|--|
| 1053 | “Java Secure Socket Extension (JSSE) Reference Guide for Java Platform Standard Edition 7”, Oracle Java SE Documentation, archived on November 25, 2011 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20111125044243/http://docs.oracle.com/javase/7/docs/technotes/guides/security/jsse/JSSERefGuide.html (“JSSE Guide”). |
| 1054 | “Java™ Remote Method Invocation API (Java RMI)”, Oracle Java SE Documentation, archived on November 27, 2011 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20111127072151/http://docs.oracle.com/javase/7/docs/technotes/guides/rmi/index.html (“Java RMI Overview”). |
| 1055 | “Using Custom Socket Factories with Java RMI”, Oracle Java SE Documentation, archived on April 3, 2012 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20120403095031/https://docs.oracle.com/javase/7/docs/technotes/guides/rmi/socketfactory/index.html (“Java RMI Socket Overview”). |
| 1056 | “Using Java™ RMI with SSL”, Oracle Java SE Documentation, archived on July 3, 2012 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20120703221253/https://docs.oracle.com/javase/7/docs/technotes/guides/rmi/socketfactory/SSLInfo.html (“Java RMI SSL Overview”). |
| 1057 | Microsoft Computer Dictionary, 3rd ed., 1997, excerpts (“Microsoft Computer Dictionary”). |
| 1058 | Usenix Jails in FreeBSD for Fun and Profit, Paco Hope, USENIX ;login: The Magazine of Usenix & Sage, vol. 27, number 3, June 2002 (“Hope”), archived on March 16, 2003 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20030316225604/http://www.usenix.org/publications/login/2002-06/pdfs/hope.pdf . |
| 1059 | Excerpts from Silberschatz et al., Operating System Concepts (8 th Ed. 2009) (“Silberschatz 2009”). |
| 1060 | Excerpts from Silberschatz et al., Operating System Concepts (9 th Ed. 2012) (“Silberschatz 2012”). |
| 1061 | Excerpts from McKusick and Neville-Neil, “The Design and Implementation of the FreeBSD Operating System” (2005) (“McKusick”). |

| <i>Exhibit #</i> | <i>Description</i> |
|-------------------------|---|
| 1062 | Excerpts from Benevenuti, “Understanding Linux Network Internals” (2006) (“Benevenuti”). |
| 1063 | Excerpts from Stallings, “Operating Systems Internals and Design Principles” (7 th ed. 2012) (“Stallings”). |
| 1064 | “apache_1.3.19.tar.gz.md5” Hash File, Apache Software Foundation, archived on June 28, 2001 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20010628043950/http://www.apache.org/dist/httpd/apache_1.3.19.tar.gz.md5 . |
| 1065 | “apache_1.3.19.tar.gz.asc” Signature File, Apache Software Foundation, archived on June 28, 2001 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20010628043146/apache.org/dist/httpd/apache_1.3.19.tar.gz.asc . |
| 1066 | “Apache Archive Distribution Directory”, Apache Software Foundation, available at https://archive.apache.org/dist/httpd (“Apache Archive”). |
| 1067 | U.S. Department of Commerce, FIPS PUB 186-2, Digital Signature Standard (DSS) (January 27, 2000) (“DSS FIPS PUB”). |
| 1068 | J. Preshing, “Hash Collision Probabilities”, Preshing on Programming, May 4, 2011, archived on September 27, 2011 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20110927200414/preshing.com/20110504/hash-collision-probabilities (“Preshing Hash Collision Probabilities”). |
| 1069 | “Avalanche Effect”, Wikipedia, archived on April 30, 2010 at Internet Archive’s Wayback Machine at https://web.archive.org/web/20100430122132/https://en.wikipedia.org/wiki/Avalanche_effect (“Wikipedia Avalanche Effect”). |
| 1070 | Plaintiff Edge Networking Systems, LLC’s Opening Claim Construction Brief, filed as Document No. 31, on December 17, 2024, in Case No. 1:24-cv-00215-DAE, captioned as <i>Edge Networking Systems, LLC v. Microsoft Corporation</i> (W.D. Texas). |
| 1071 | Defendant Microsoft Corporation’s Responsive Claim Construction Brief, filed as Document No. 32, on January 17, 2025, in Case No. 1:24-cv-00215-DAE, captioned as <i>Edge Networking Systems, LLC v. Microsoft Corporation</i> (W.D. Texas). |
| 1072 | Scheduling Order, filed as Document No. 27, on September 4, 2024, in Case No. 1:24-cv-00215-DAE, captioned as <i>Edge Networking Systems, LLC v. Microsoft Corporation</i> (W.D. Texas). |

I. INTRODUCTION

Pursuant to 35 U.S.C. §§311 *et seq.* and 37 C.F.R. §42.1 *et seq.*, Microsoft Corporation (“Petitioner”) petitions for an *inter partes* review (“IPR”) of U.S. Patent No. 11,695,823 (“’823 Patent”). Petitioner submits that ’823 Patent Claims 1-5, 7-8, 12-15 and 18-19 (the “Challenged Claims”) are unpatentable under 35 U.S.C. §103 in view of the prior art references discussed herein. This Petition demonstrates by a preponderance of the evidence that there is a reasonable likelihood that Petitioner will prevail with respect to at least one of these claims. Accordingly, Petitioner requests that the Board institute an IPR of the ’823 Patent pursuant to 37 C.F.R. §42.108.

II. OVERVIEW

Throughout this Petition, the following color scheme is used to annotate the same elements in the ’823 Patent and the prior art¹:

| Concept | Color |
|---|------------------------------|
| distributed system comprising a network device and a cloud device | alternating green and maroon |
| network device | green |
| cloud device | maroon |

¹ Petitioner has added all emphasis in quoted materials and annotations to the figures.

| | |
|--|-----------------------------|
| distributed application with a component on a network device and a component on a cloud device | alternating pink and orange |
| network device application that is a component of the distributed application hosted on a network device | pink |
| cloud application that is a component of the distributed application hosted on a cloud device | orange |
| application management portal (and associated software) | blue |
| infrastructure application marketplace | brown |

The Challenged Claims recite a system including programmable network and cloud devices spread across a network (e.g., the Internet), a distributed software application with components installed on each of these devices, application programming interfaces (APIs) to program these components independent of these devices' hardware, an "application management portal" "managing lifecycles of the distributed applications", and an infrastructure application marketplace providing these distributed applications.

The '823 Patent uses the term "distributed software-defined network (dSDN)" to describe a system with this functionality. Annotated FIG. 3 illustrates one embodiment of a dSDN:

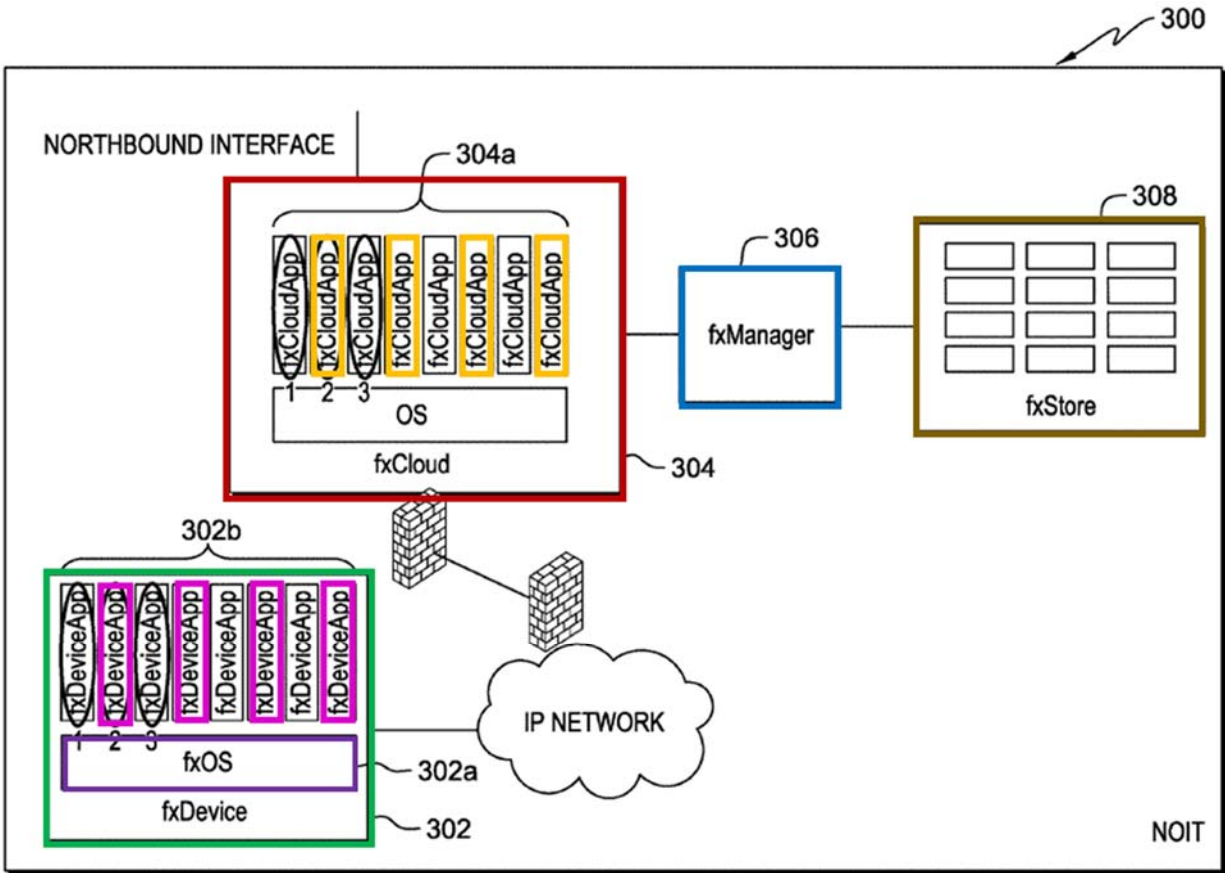


FIG. 3

Figure 3 shows “distributed applications” 1, 2, and 3 (among other distributed applications), and the two components of distributed applications 1, 2, and 3 respectively installed on a “network device” and a “cloud device.” Figure 3 also shows a network, labeled “IP Network” across which the “network device” and “cloud device” are distributed, an “application management portal” to manage the lifecycle of the distributed applications, and an “infrastructure application marketplace” that provides applications to the “application management portal” “for installation in” the “network device” and “cloud device.”

The Challenged Claims, when stripped of their jargon, claim a system including (i) various **devices** spread across a network, (ii) a distributed **software application** having components installed on these **devices**, and (iii) a **management interface** and **repository** that allows a user to download, install and uninstall the components of the distributed **software applications**.

This basic design was well-known and widely implemented prior to June 2013. Vasell (EX1004/EX1015/EX1016), filed fourteen years before the '823 Patent's priority date, provides an example of such a **distributed system** and distributed **software applications**. Like the '823 Patent, Vasell discloses a system having **devices** spread across the Internet, called a "**service gateway system**." Vasell discloses a "**network device**" called a "**service platform server**," and a "**cloud device**," called "**service provider equipment**" or a "**network operator server**." Vasell discloses a plurality of distributed **software applications**, each having a **component** installed on a **network device** and a **component** installed on a **cloud device** (and each implementing a service for end users). And Vasell discloses a **management service** having an **external interface** to download, install and uninstall the distributed **applications**. Annotated FIG. 2 of Vasell shows one embodiment of a "**service gateway system**":

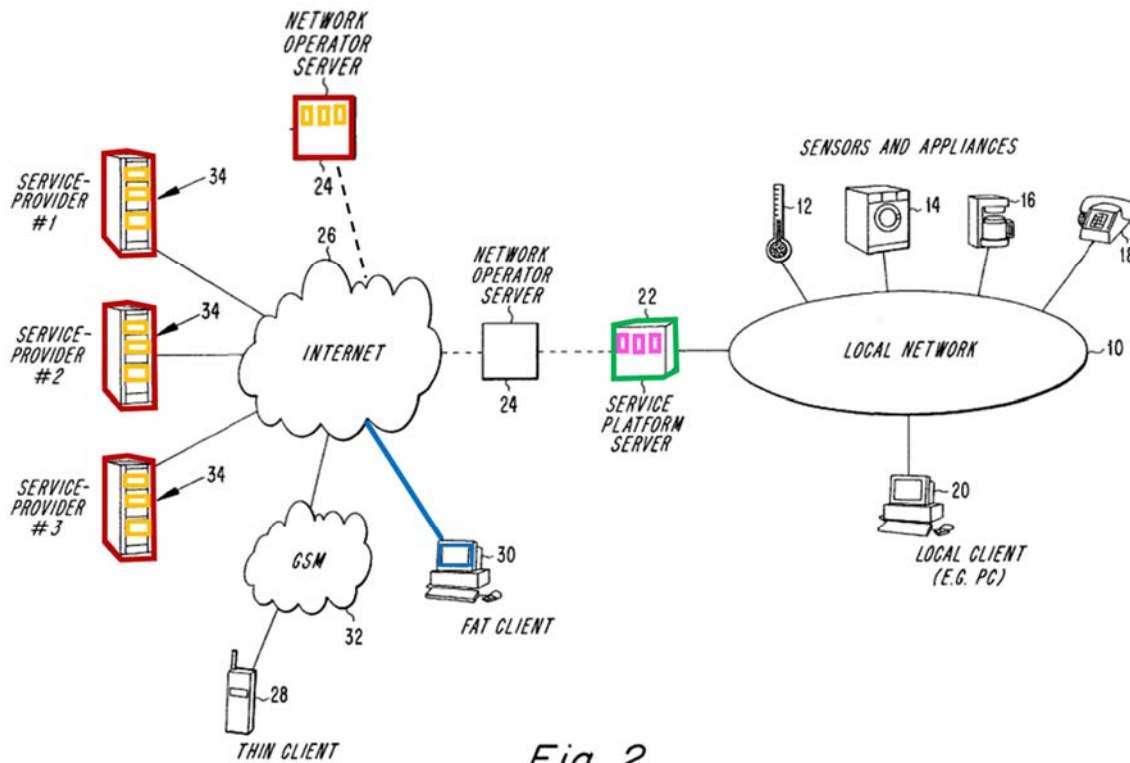


Fig. 2

Unsurprisingly, Vasell leaves certain implementation details to design choices of POSITAs. However, Vasell’s architecture and methods were foundational concepts (including implementation in a Java run-time environment) that were eventually standardized under the Open Standards Gateway initiative (OSGi) specifications. Technical implementation details that were intended to implement Vasell’s teachings were described in each of the published references cited in Grounds 1 and 2—Alves (EX1008), Hall (EX1009) and Rellermeier (EX1011). Accordingly, POSITAs would naturally combine the teachings in these references together, and their combined teachings disclose each limitation recited in the Challenged Claims.

III. GROUNDS FOR STANDING (37 C.F.R. §42.104(A))

Petitioner certifies the '823 Patent is available for IPR and Petitioner is not barred or estopped from requesting an IPR of the Challenged Claims on the grounds identified herein. 37 C.F.R. §42.104(a). This Petition is filed pursuant to 37 C.F.R. §42.106(a).

IV. RELIEF REQUESTED (37 C.F.R. §42.22(a))

Petitioner respectfully requests institution of an IPR pursuant to 37 C.F.R. §42.108 and cancellation of the Challenged Claims.

V. REASONS FOR THE REQUESTED RELIEF

As explained in §II and §VI-§VIII of this Petition and in the attached Declaration of Petitioner's Expert, Dr. Erez Zadok, ("Zadok"), EX1003, the **distributed systems** implementing distributed **applications**, described and claimed in the '823 Patent, were obvious over the prior art to a person of ordinary skill in the art ("POSITA") in June 2013.

A. '823 Patent Summary

The Patent and Challenged Claims are directed to a system that the inventors called a "**distributed software-defined network (dSDN)**." FIG. 3 (below) illustrates "a high-level overview of a **dSDN** system 300":

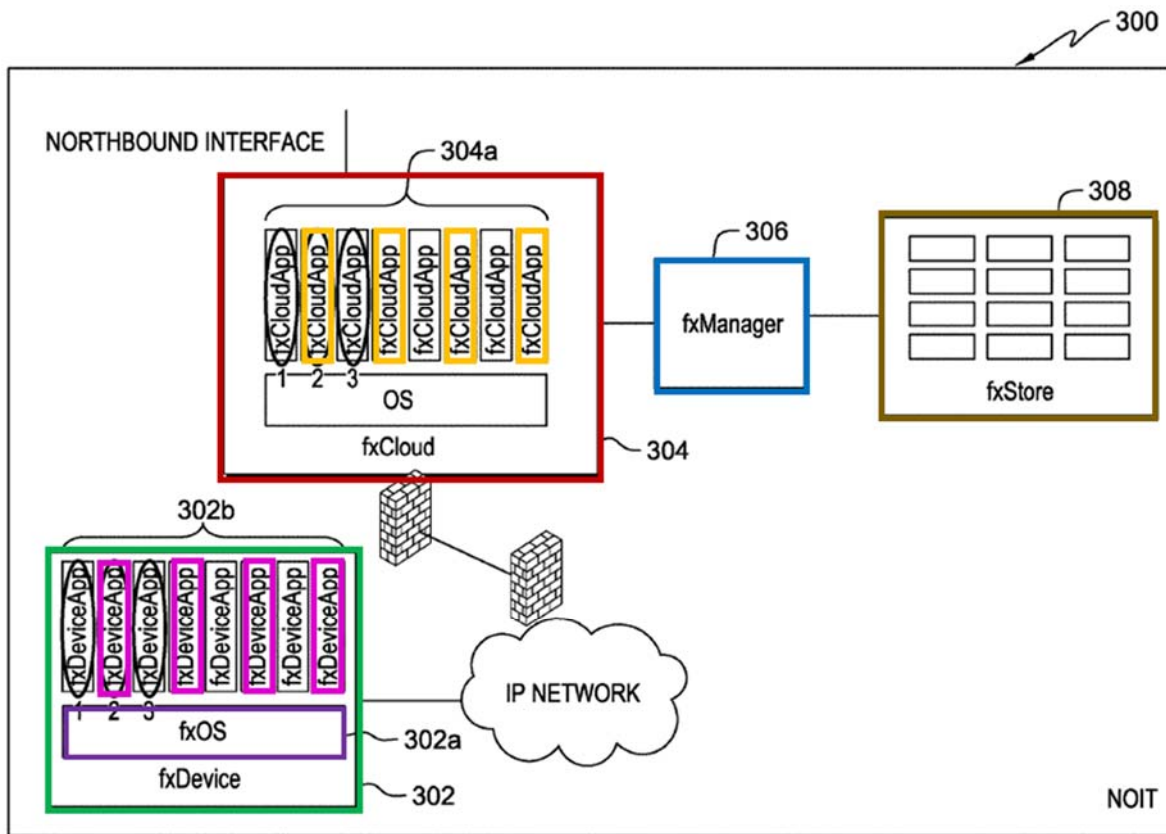


FIG. 3

EX1001, 10:8-13.

The exemplary dSDN system includes two hardware devices: “flexible network device (fxDevice)” and “flexible cloud platform (fxCloud).” The fxDevice is “any networking equipment” that is “powered by a sandboxing operating system named flexible operating system (fxOS).” EX1001, 10:13-15. The fxCloud is “backend cloud infrastructure” (e.g., one or more “network servers”), which could be a private, or “shared”/public, cloud platform. EX1001, 10:26-27, 11:32-33, 24:5, 25:36-37, 3:17-19, 17:62-66, 25:3-7.

Beyond the hardware components of the **system**, the Patent also described the distributed **applications** (e.g., 1, 2, 3 in FIG. 3 (above)) that run across these **devices**. Each distributed **application** can be separated into a **component** installed on an **fxDevice** (called “**fxDeviceApp**”) and a component installed on an **fxCloud** device (called “**fxCloudApp**”). EX1001, 10:21-31. For example, Figure 12 describes “an example procedure for Application (dApp) Provisioning.” EX1001, 18:47-48. The **administrator** identifies a distributed **application** at the start. The system then identifies components of that **application** that should be installed on the **fxDevice** and **fxCloud** device, and installs a respective component on each device. EX1001, 19:8-28.

In other words, a distributed **application** includes an **fxCloudApp component** “paired with” an **fxDeviceApp component**. EX1001, 10:21-31; EX1003, ¶¶243-246.

The Patent describes the “Virtual Fabric (fxVF)” (annotated gray below) **dSDN** “service.” The virtual fabric “provides an abstraction layer for applications to communicate with each other whether they are in the **fxDevice** or **fxCloud**”, and “enables transparent switching of application and system messages of a dApp between the **fxCloud 304** and the **fxDevice 302** and between different dApps”:

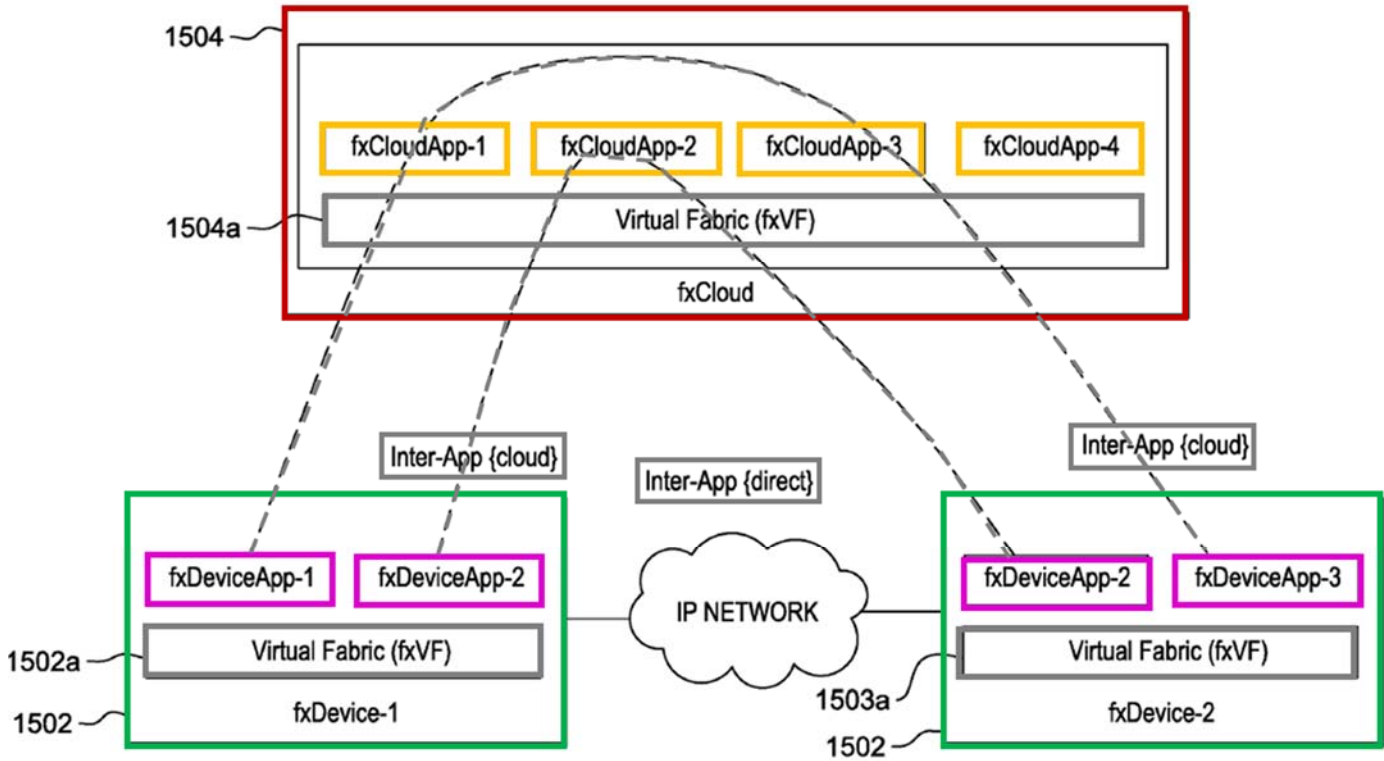


FIG. 15

EX1001, 20:35-49, 13:39-42, FIG. 15 (above), 10:40-46; EX1003, ¶¶250-253.

The exemplary dSDN system 300 includes an “[application management portal \(fxManager\)](#).” The [fxManager](#) “presents a user-friendly portal to [a] network administrator”, through which the administrator may, for example, manage the [fxDevice\(s\)](#), manage the [fxCloud server\(s\)](#), perform “[a]pplication lifecycle [management](#)” tasks (e.g., “provisioning, de-provisioning, upgrade/downgrade applications”), and/or “brows[e] through the certified applications in the [fxStore](#).” EX1001, 15:10-16:27, 10:47-51, 17:53-19:28.

The Patent describes that, using well-known digital signature techniques, an application could be “signed” (so that, if a signature match is obtained by the

recipient (*e.g.*, [fxManager](#)) using these same techniques, the application is certified to be authentic and intact (unchanged). EX1001, 23:21-24:11; EX1003, ¶¶258-264.

The exemplary [dSDN](#) system 300 includes an “[infrastructure application market place \(fxStore\)](#).” The [fxStore](#) may be “third party hosted”, contains software applications, and associated updates, developed (and optionally tested) for use in the [dSDN](#) environment by application developers, and from which a network administrator may download such applications and updates (*e.g.*, via [fxManager](#)). EX1001, 10:12-13, 10:51-55, 13:1-3, 15:39-40, 15:48-49, 16:25-35.

B. Prosecution History

The Challenged Claims received only double patenting and written description rejections, which were overcome by a terminal disclaimer and amendments deleting the word “simultaneously” from the claims. EX1002, 304-311, 332-336, 444-464.

C. Claim Construction

Petitioner proposes that each claim term in the Challenged Claims be given its plain and ordinary meaning in this proceeding, and that no specific construction of any claim term is required because the prior art relied on in this Petition meets each of the claim terms under any reasonable construction. EX1003, ¶367. The parties have identified and briefed different constructions for various claim terms

pending in the Related Litigation. *See* EX1070; EX1071. However, the Petition Grounds render the claims unpatentable under either party’s construction, and any reasonable construction, and thus the Board need not construe any term.

Petitioner addresses the below terms of the Challenged Claims to the extent Patent Owner (“PO”) argues or the Board finds they are governed by means-plus-function. Solely for purposes of this Petition, for each term, this relies on PO’s identification (in EX1070) of “specific portions of the specification that describe the structure” as corresponding to the recited function. *Id.*; EX1003, ¶¶368-374.

- “virtual fabric” (claim 12): “an abstraction layer for applications to communicate with each other” as described in EX1001, 13:33-39, 20:36-41, 11:57-62. EX1070, 9-10; EX1071, 12-13.
- “load controller adapted to monitor loads ... and affect change in accordance with thresholds received ...” (claim 7): “a controller executing at least one of a ‘cloud breathing’ or ‘load monitoring’ application” as described at EX1001, 21:66-22:5, 22:20-22 (load monitoring), 15:64, 22:9-26, FIGS. 17A-17B (cloud breathing). EX1070, 16-19; EX1071, 19-20.

D. Priority Date of the Challenged Claims

Petitioner assumes the ’823 Patent’s earliest-claimed priority date—June 13, 2013—is the priority date.

E. Person of Ordinary Skill in the Art

A POSITA 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. EX1003 ¶58.

F. State of the Art

The following section describes the state of the art for distributed systems and virtualization technologies as of June 2013. The prior art references, and the discussions of what was known to POSITAs, provide the factual support for the general description of the state of the art at the time of the invention, and provide additional motivation to modify or combine the references. Accordingly, these references should be considered by the Board.

1. Sandboxing Operating Systems

Traditional computer operating systems (“OS”) (*e.g.*, Linux, Unix, etc.) included mechanisms to “sandbox” software programs (*e.g.*, confining and restricting what resources a software application can access and when). EX1059, 591; EX1058, 2; EX1003, ¶¶115-117. Additional “sandboxing” was well-known

to be achieved through (i) virtualization of the entire OS, which allowed multiple instances of the same OS to be run simultaneously on the same hardware, or through (ii) virtual machine (“VM”) technology, which was more than 40 years old as of June 2013, and which allowed the same hardware to host multiple VMs, each potentially running a different OS. EX1058, 3; EX1031, 569-570; EX1057, 498; EX1061, 124-127; EX1052, 100-103; EX1003, ¶¶98, 111, 118-119, 124-127, 218.

VM technology provided the benefit of isolating programs running in different OSs and permitted companies with thousands of servers doing many different tasks to have fewer physical machines. This represented costs savings, facilitated software development, and allowed for load balancing across VMs on different servers. EX1031, 569-570; EX1057, 498; EX1003, ¶¶98, 111.

As an example, in 1991, Sun Microsystems created the “Java” software language to provide “a method to execute programs without any platform dependence” and whose “popularity soared in 1994 when it was used across the Internet.” EX1034, 304. 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 a small, isolated area called the “sandbox.” Specifically, the well-known “Java sandbox” of the JVM provided a very restricted environment in which to run untrusted software code obtained from open sources (restricting its

access to the limited resources provided inside the sandbox) alongside trusted code from trusted sources (which could have full access to system resources). EX1034, 304-305, 406; EX1045, 44-45; EX1044, 642-643; EX1009, 438; EX1003, ¶¶129-134.

2. Security for Software Downloaded/Implemented over the Internet

Challenged Claim 18 recites “verif[ying] the authenticity and integrity of updates to the... applications.” This was also a well-known security tool for software downloaded/implemented over the Internet.

For example, one important security aspect of Java software (“JAR files”) and JVMs was the use of digital signatures. A “digitally signed” Java application, was created and delivered in Java format using standardized techniques, and was treated as if it was trusted software code if the “public key”, available in the signer’s (*e.g.*, application developer’s) digital certificate, was usable by the application recipient to successfully “decrypt” the signature (affirming that the matching private key, of the public-key-private-key-pair unique to the signer, signed the application in the first instance). EX1045, 169-172; EX1057, 432, 131, 145, 388-389; EX1067; EX1003, ¶¶81-84, 87, 136-138.

Java’s use of “digital signatures” was widely implemented so that recipients/implementers of the application could verify the “authenticity” and “integrity” of the application, and of application updates, “when...downloaded”

and “before... install[ing] it.” *Id.*; EX1044, 590-591, 641-644; EX1045, 265-266; EX1050, 8-10; EX1003, ¶¶139-147.

3. Distributed Systems

The term “distributed system” has long been known to POSITAs as “a software system built on top of a network”, where “the distinction between a network and a distributed system lies with the software (especially the [OS]), rather than the hardware”, and where the “user of a distributed system is not aware that there are multiple processors, it looks like a virtual uniprocessor.” EX1032, 2; EX1057, 153-154; EX1003, ¶¶148, 151-152.

Software “modularity”—where “the code of [a single] software application is divided into logical parts representing separate concerns”—was long understood as an important concept for building a software system on top of a network, because modular software application solutions (with software components distributed across multiple network devices) provided isolation, made software development more scalable, allowed parallel development of application-modules/components, and led to well-defined application programming interfaces (APIs). EX1009, 4-5; EX1008, xiii; EX1063, 689; EX1052, 46-47, 271; EX1003, ¶¶149-150, 164-166.

The evolution of distributed systems has been taught to computer scientists for years. EX1003, ¶155. As soon as computers could communicate with each

other, networks of computers began to evolve incrementally over the years—from simple client/server configurations, to more complex computer “clusters”, to managed private/public “clusters” nowadays called clouds. EX1063, 677-712; EX1052, xxv, 19, 45, 272; EX1057, 153; *see* §V.F.3.b; EX1003, ¶¶151, 156-160.

a. Open Standards Gateway Initiative (OSGi)

A well-known initiative related to modular software applications and **distributed systems**—the Open Standards Gateway Initiative (OSGi)—commenced with a three-year joint research project in the late 1990s between Swedish companies Ericsson and CR&T, from which, in 1999, a patent application describing foundational concepts (“Vasell”, EX1015/EX1016/EX1004, *see* §VII.A.1, §VII.A.5-§VII.A.6) was filed, and the non-profit OSGi alliance was founded. EX1015; EX1016, 3-9; EX1003, ¶¶177-188.

Over the succeeding years, technical implementation details were developed and published in the OSGi’s open specifications (*e.g.*, EX1028) and numerous OSGi-related textbooks (*e.g.*, EX1009, EX1008), whitepapers (*e.g.*, EX1013, EX1050) and articles (*e.g.*, EX1029, EX1010, EX1012, EX1011). EX1003, ¶189.

Vasell depicted a **distributed system** “end-to-end architecture” using JVMs hosted on respective **devices**’ OSs, which was used as the foundational architecture for the first OSGi specification in January 2000. Illustrated below is FIG. 2 of

Vasell (top) and FIG. 2 from the OSGi specification (bottom), each showing hardware devices spread across the Internet:

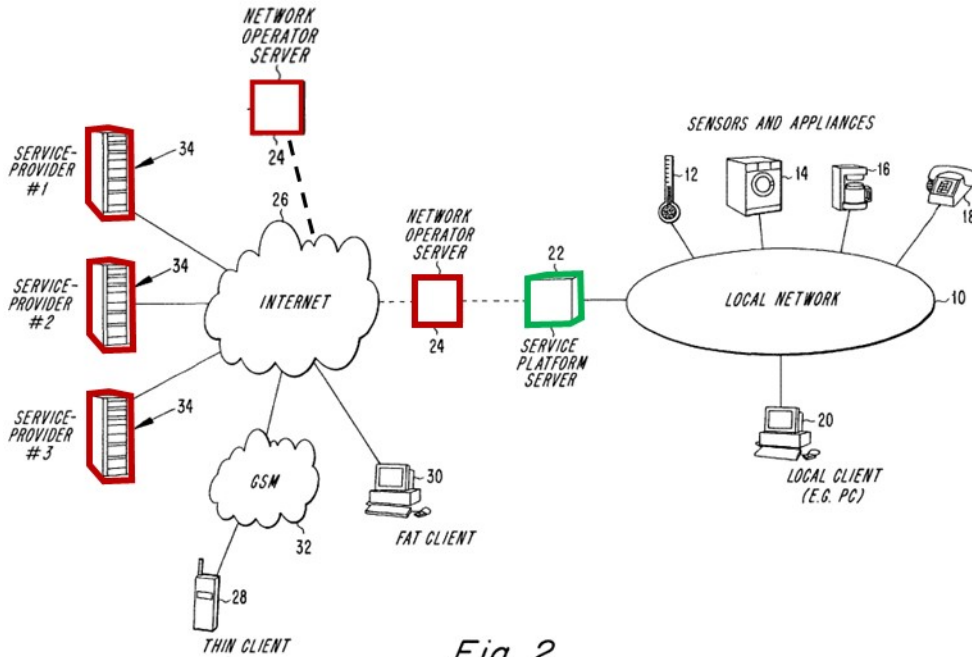


Fig. 2

EX1004, FIG. 2.

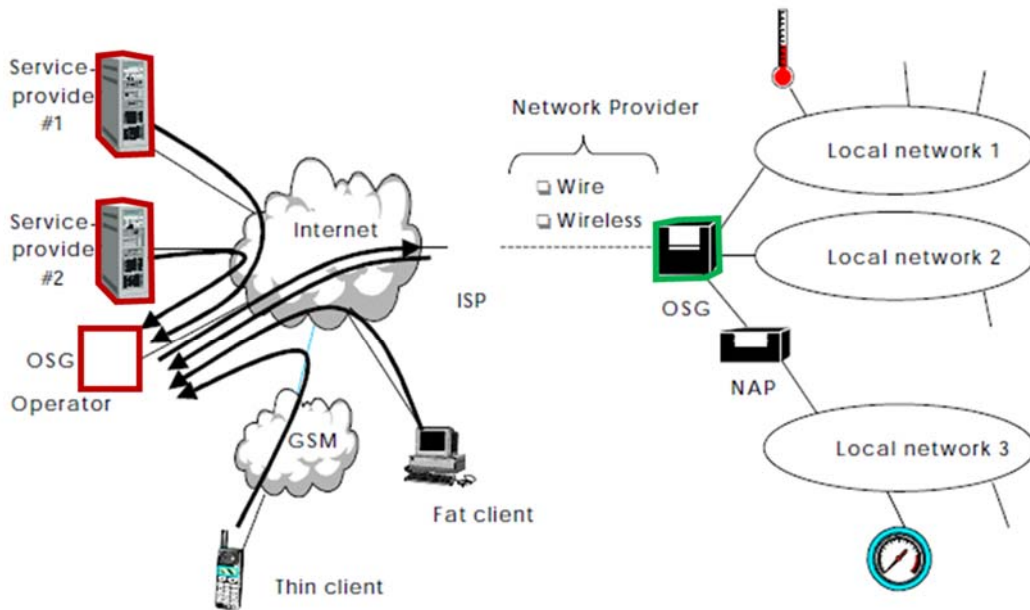
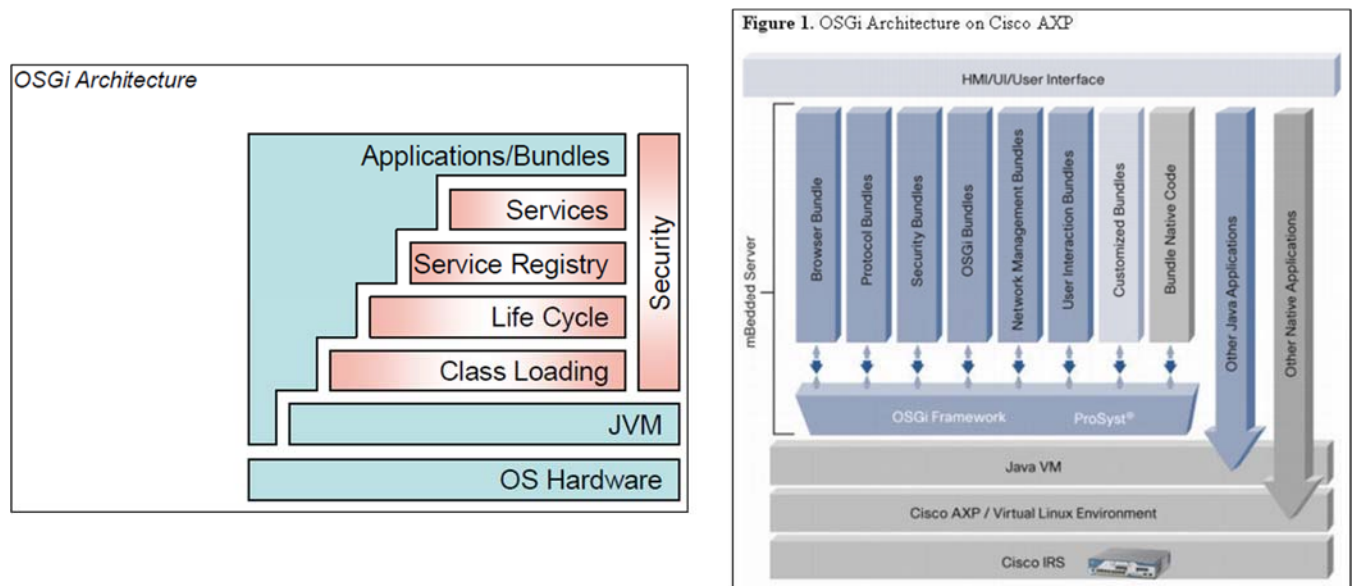


Figure 2. OSGi End to End Architecture

EX1050, FIG. 2; EX1003, ¶190.

The standardized “OSGi Service Platform” provided a “portable and secure execution environment” based on “the Java Runtime Environment (JRE), which include[d] the [JVM] that isolates the software component developer from the details of the underlying OS and hardware.” EX1008, 2, 6-7, 16. The platform was used to “dynamically install and uninstall Java modules, which OSGi call[ed] ‘bundles.’” *Id.* Accordingly, the standardized OSGi architecture was virtualized on a device (*e.g.*, router, server, etc.), executing in at least one JVM running on a host OS (*e.g.*, Linux, UNIX, etc.) of the device:



EX1013, 3-4; EX1029, 1-2; EX1010, Figure 2; EX1016, 7; EX1003, ¶¶191-192.

The “OSGi [remote management](#) architecture” included a “[management API](#)” (the claimed “[application management portal](#)”) that allowed the installation of new bundles (and associated updates) in a remote JVM (and uninstalling them),

including the use of “standardized **hot update** technology” that allowed runtime updates to multi-bundle applications, “without restarting the JVM.” EX1013, 7, 10; *id.*, 3-4, 11, 17-18; EX1003, ¶¶197-199.

The OSGi-standardized platform also utilized a “**bundle repository**” (the claimed “**infrastructure application marketplace**”) that contained OSGi bundles developed by the OSGi Alliance and third party developers, and, for OSGi framework administrators, provided a “**web based interface**” through which the administrator could search for, download, and install bundles. EX1041, 1; EX1013, 7-8, 14-15; EX1051, 1; EX1042, 1-2; EX1003, ¶¶200-203.

OSGi’s standardized Enterprise implementation (EX1028) included communications between a **locally-installed** “**bundle** that’s exporting a remote service in one OSGi framework instance and a **remotely-installed bundle** that’s importing the service in another [OSGi] framework instance” (the claimed “**distributed application**” with components installed on each **device**) using well-known protocols for providing secure communications between **software** components of a **distributed application** such as RMI, SOAP and REST. EX1008, xiv, 249, 250, 252-255, 261-263, Figures 10.9-10.10; EX1046, 1-2, 7-8, 13-15; EX1047, 1-2, 4; EX1048, 1-2; EX1053, 3-5, 68; EX1054, 1; EX1055, 1; EX1056, 1; EX1003, ¶¶205-212.

POSITAs also understood that “the OSGi platform is an ideal platform for cloud computing.” EX1008, 250, 269, xiii-xiv; EX1011, 3; EX1003, ¶213.

b. Cloud

As discussed in §V.F.3, “clouds” was the name adopted in the mid-2000s for managed private/public computer “clusters”, and were simply the latest, well-known evolution of **distributed systems**. EX1063, 699-708; EX1052, xxv-xxvi, 19-20, 45, 271-272; EX1003, ¶¶157-160, 169-176, 214.

POSITAs understood a “cloud” to mean a “set of computers accessed remotely”, or “an infrastructure comprising a network of interconnected nodes”, with the “key concept” being that “the user does not know where the server is located, or where there is one server or many; everything is ‘off in the clouds’.” EX1033, 94; EX1052, xxv, 3, 45; EX1030, [0038]; EX1057, 153; EX1003, ¶215. Relatedly, POSITAs understood “cloud computing” to mean “computing operations carried out on servers that are accessed through the Internet” or “applications and services that run on a distributed network using virtualized resources and accessed by common Internet protocols and networking standards.” EX1033, 94; EX1052, 3; EX1003, ¶216.

Cloud deployment models generally included private, community (shared by several organizations), public and hybrid (two or more different types of clouds). EX1052, 3, 7-8; EX1030, [0033]-[0037]; EX1003, ¶217.

The 40+ year-old “virtualization” technology (including as implemented through VMs (*e.g.*, JVMs)) was ubiquitous in **cloud servers** by June 2013. §V.F.1 (EX1031, 569-570; EX1036, 278-279); EX1052, xxvi, 10-12, 19-20, 51-52, 94-95, 100-103; EX1003, ¶218. Having a plurality of VMs running on a single **cloud server**—*i.e.*, on a single **server** of a set of servers accessed remotely and/or within a network of interconnected nodes—was well-known to POSITAs long before June 2013. *Id.*; EX1030, [0033]-[0037]; EX1052, 69-70, 92; EX1012, 19-22, FIG. 1; EX1043, 124-125; EX1039, 1-3; EX1003, ¶219.

It was also well-known to utilize “**application repositories**”/“**application stores**” (the claimed “**infrastructure application marketplace**”) in cloud-based **distributed systems**, including those implementing OSGi technology (one example depicted below):

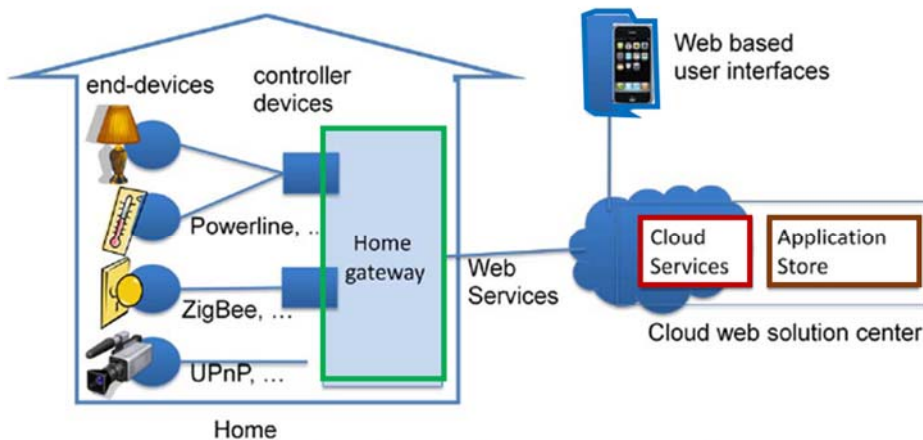


Figure 1: High-level architecture for the smart home

EX1010, 206, 208-209, 212, Figures 1-2; EX1030, [0057], FIG. 4, [0070]-[0074], [0082]-[0084]; EX1042, 1-2; EX1003, ¶¶224-226.

VI. IDENTIFICATION OF CHALLENGES

A. Challenged Claims

'823 Patent Claims 1-5, 7-8, 12-15 and 18-19 are challenged in this Petition.

B. Statutory Grounds for Challenges

The Challenges are set forth in detail below and summarized as follows:

| Ground | Claims | Basis | Reference |
|--------|-----------------|-----------|--|
| 1 | 1-2, 12-15, 19 | §103(AIA) | Vasell (EX1004/EX1015/EX1016) in view of Alves (EX1008) and Rellermeyer (EX1011) |
| 2 | 3-5, 7-8 and 18 | §103(AIA) | Vasell in view of Alves, Rellermeyer, and Hall (EX1009) |

Vasell is Patent 6,496,575, issued December 17, 2002 (EX1004), which claims priority to, while expressly incorporating by reference, Provisional Applications 60/088,437 filed June 8, 1998 (EX1015) and 60/123,971 filed March 12, 1999 (EX1016). EX1004, 1, 1:5-10. Accordingly, Petitioner refers to the combined disclosure of EX1004, EX1015 and EX1016 as “Vasell”, and makes each citation to the corresponding Vasell exhibit.

Alves (EX1008) is “OSGi in Depth”, by Alves, published in December 2011, publicly accessible by March 2012, and actually disseminated to interested persons at least by November 2012. EX1014, ¶¶44-65.

Rellermeyer (EX1011) is “Dependability as a cloud service: a modular approach,” by Rellermeyer et al., presented at the 2012 IEEE/IFIP International

Conference on Dependable Systems and Networks Workshops, published in the conference proceedings by IEEE, in August 2012, publicly accessible by late-September 2012, and actually disseminated to interested persons at least by April 2013. EX1014, ¶¶20-43.

Hall (EX1009) is “OSGi in Action”, by Hall, et al., published in April 2011, publicly accessible by December 2011, and actually disseminated to interested persons at least by December 2011. EX1014, ¶¶66-87.

Each reference qualifies as prior art under AIA §102(a)(1), and none of these references were cited nor applied by the examiner during the '823 Patent prosecution.

VII. IDENTIFICATION OF HOW THE CHALLENGED CLAIMS ARE UNPATENTABLE

A. Prior Art in the Grounds

1. Vasell

Vasell discloses foundational concepts of a **distributed system** architecture, including implementation in a Java run-time environment, that was eventually standardized under OSGi. §V.F.3.a; EX1003, ¶¶177-190, 299. Vasell first describes the hardware of “the **service gateway system**,” which is its system for processing data packets. The **service gateway system** is shown in Vasell’s Figure 2 as annotated below (“Annotated FIG. 2”). The hardware includes (i) a **service platform server 22** connected to the local area network 10 and connected to (ii) a

network operator server 24 “via the Internet”, which is connected to a (iii) plurality of service provider equipment 34 “via... Internet 26”:

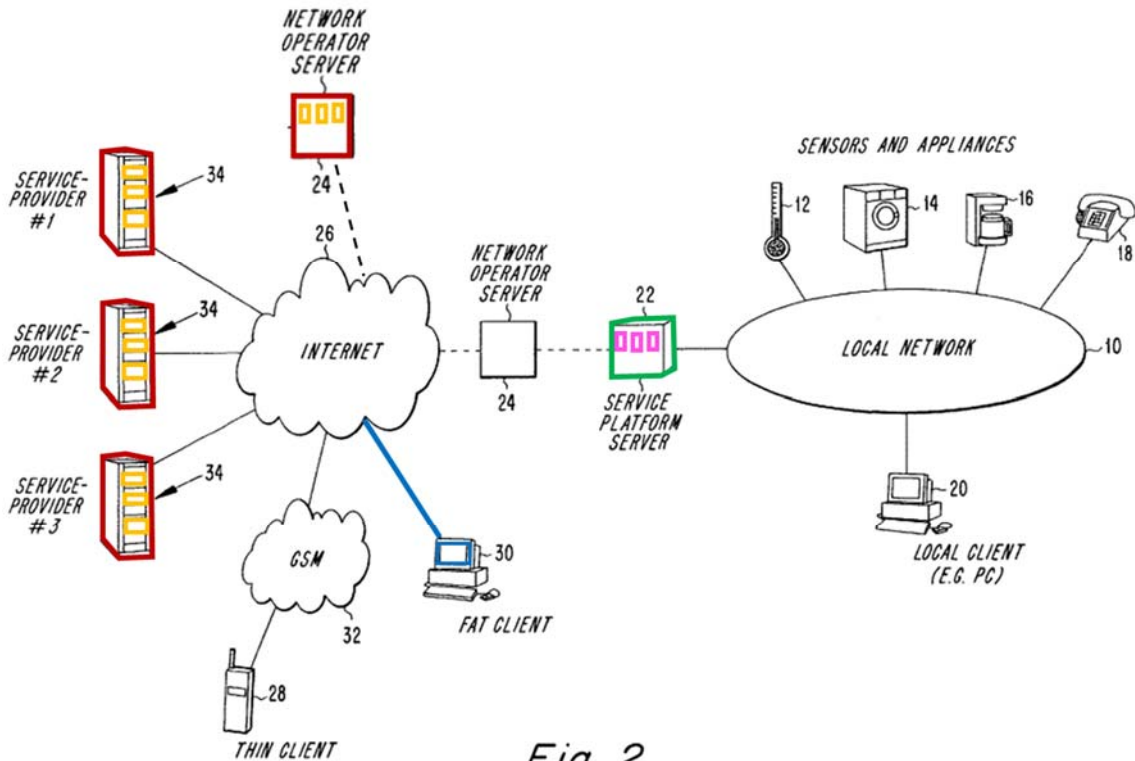


Fig. 2

EX1004, 4:31-47, 5:61-6:20 (“[s]ervice platform server 22” is “an edge server”), 6:62-7:6 (“[n]etwork operator server 24 equipment... includes one or more servers”), 7:14-18 (“[s]ervice provider equipment 34... includes computer systems or servers”), 7:12-14; EX1016, 4, 6-9, FIGS. 2, 5; EX1003, ¶¶302-303, 306-310.

Vasell also discloses a plurality of distributed software applications—called “service applications”, each having a component installed on a service platform server and a component installed on a network operator server and/or service provider equipment, and each implementing a “service” for end users. End users

ranged from “household users and residences..., an entire office building, an industrial plant, or even a campus-type organizational facility.” EX1004, 10:32-41; EX1003, ¶314.

Annotated FIG. 2 shows, as disclosed in embodiments in Vasell, a plurality of **service applications** (each including a plurality of **distributed** components). Specifically, each component of a **service application** is hosted on a JVM running on the respective OS of a **service platform server 22**, **network operator server 24**, or **service provider equipment 34**. *Id.*; EX1004, 2:64-3:1, 8:53-9:18, 24:49-51, 12:27-29, 11:13-58, 18:23-30; EX1016, 4, 7-9; EX1003, ¶¶304-305.

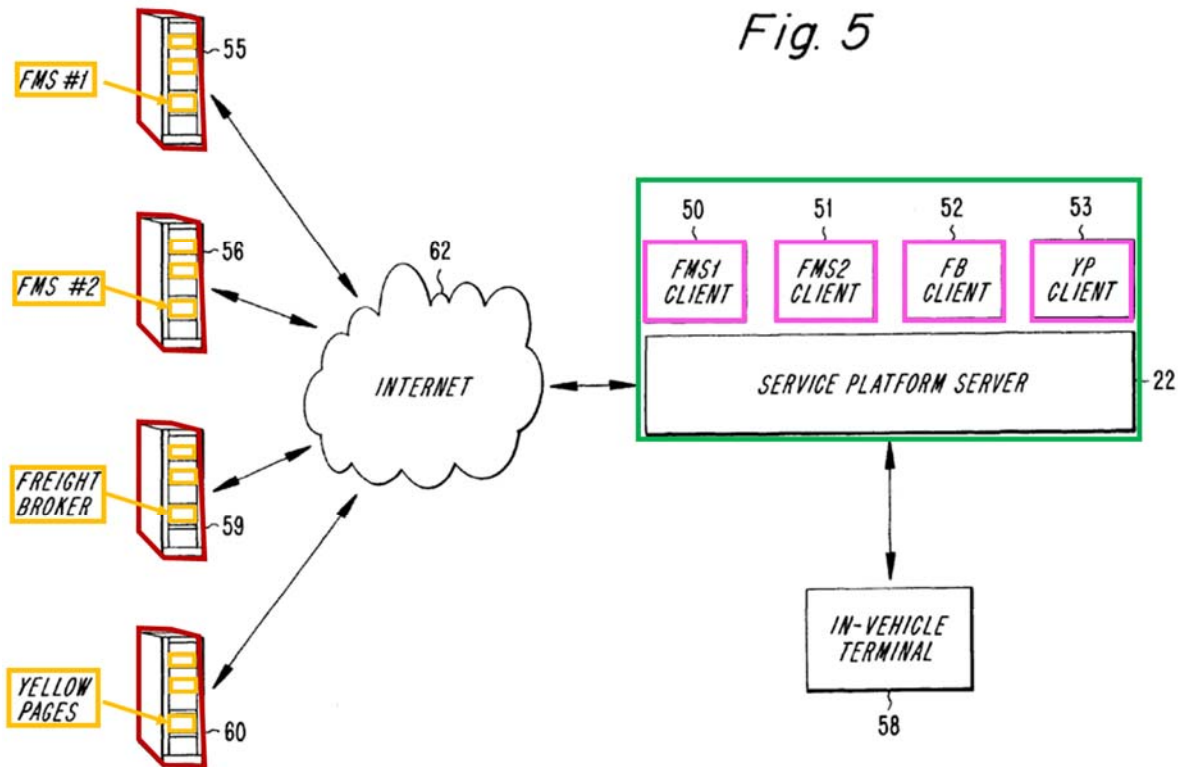
POSITAs understood Vasell’s disclosed network of **network operator server(s) 24** and **service provider servers 34**, accessed “transparent[ly]” through the Internet, each respectively hosting distributed **service application** components on a JVM running on the OS on the **server’s** hardware to implement services through the **service gateway system**, to constitute a “cloud”, and each such **server** a programmable **cloud device**. EX1004, 2:60-3:1; §V.F.3.b (EX1033, 94; EX1052, xxv, 3, 45-46, 271-272; EX1030, [0038]; EX1057, 153); EX1003, ¶311.

Regarding the implementation of services through the **service gateway system**, Vasell discloses that “implementation of connectivity based services according to the present invention may take advantage of **distributed processing techniques** whereby a service may be implemented using **software** operating on

two or more of the servers or processors associated with the [service gateway system] architecture illustrated in FIG. 2.” EX1004, 8:53-58, 2:60-3:1; EX1016, 4, 7-9. “For example, a first portion of software associated with [an] environmental service... may reside permanently on the #1 service provider equipment 34. A second portion of the software used to implement [this service]... may reside as part of the network operator server 24. Likewise, a third portion of the software used to implement this service may reside on the service platform server 22.” *Id.*; EX1004, 8:53-9:18, Annotated FIG. 2, 24:49-51, 12:27-29, 11:42-58; EX1016, 4, 7-9; EX1003, ¶¶312-315.

Another example of distributed service applications, where each distributed application has a component at a client location, and a component at a service provider location, is illustrated in Annotated FIG. 5. Vasell discloses mobile service platform server 22 has various “service software client packages” (50-53) loaded thereon that communicate with associated software modules loaded on service provider servers (55, 56, 59, 60) to provide services to trucking industry users:

Fig. 5



EX1004, 11:11-20, 11:34-38, 18:27-30; EX1016, 4, 7-9; EX1003, ¶316. For example, two different freight companies, FMS #1 and FMS #2, may each use mobile service platform server 22 for services by having respective software on respective service provider servers (55, 56, 59, 60) communicate with associated software packages (50-53) running on the mobile server to coordinate operations of the truck and for inter-application information exchanges (e.g., client software package may provide freight company server software with information regarding the current location of the truck and its anticipated arrival time of delivery, freight broker/yellow pages server software may provide information required by client

software packages, etc.). *Id.*; EX1004 11:17-39, 11:50-58, 2:64-66; EX1003, ¶317.

Vasell describes the service application development environment of the service gateway system as an “open” environment that included “**different** service providers” and “**third party** developers”, and “potentially opens **third-party markets** for... services.” EX1004, 3:10-15, 5:27-33, 10:18-23, 13:37-43, 22:35-37; EX1016, 4, 7-9; EX1003, ¶¶319, 332-333. Accordingly, Vasell discloses that, “mechanisms for **authentication, authorization** and access control are integrated within the service gateway system design.” *Id.*; EX1004, 4:52-65, 15:46-47.

Vasell emphasizes the importance that communications between the respective distributed service application components (“portions”/“boxlets”), each hosted on respective JVMs running on the respective OSs of the local and remote servers of the service gateway system, “**be secure**” and, therefore, expressly envisioned “**widespread** usage of... security techniques **including data encryption, various authorization and identity verification** techniques, etc. to be provided **so that the implementation, monitoring and billing associated with these services is robustly defended against intrusion and manipulation.**” EX1004, 11:64-12:10; EX1016, 3-4, 6-9; EX1003, ¶¶318-319.

Vasell discloses that the design of its modular, virtualized distributed system “ensure[d] the security, isolation and robustness of the service applications”,

“achieve[d] application integrity”, allowed “multiple **service applications**... to simultaneously operate without detrimentally affecting each other”, and “prevent[ed] operation of a **service application** from disrupting other **service applications**, thus ensuring integrity of the **service gateway system**.” *Id.*; EX1004, 20:49-52, 18:12-13, 4:58-62, 5:11-15, 18:2-5, 12:47-54, 2:50-53, 3:19-24; EX1016, 4, 6-8; EX1003, ¶¶323-324.

Vasell discloses that the **service gateway system** is “**remotely managed**”, for example, via “**management system service 76**” of “system services layer 110”, which is “implemented in the form of the **boxlets**... executing within cells”, and “provides **an external interface** through which the **service applications**... may be downloaded, installed, removed, executed, and controlled”, which are **management operations** that Vasell describes, in detail, as being performed by “**cell manager 93**” using “**cell table 94**”:

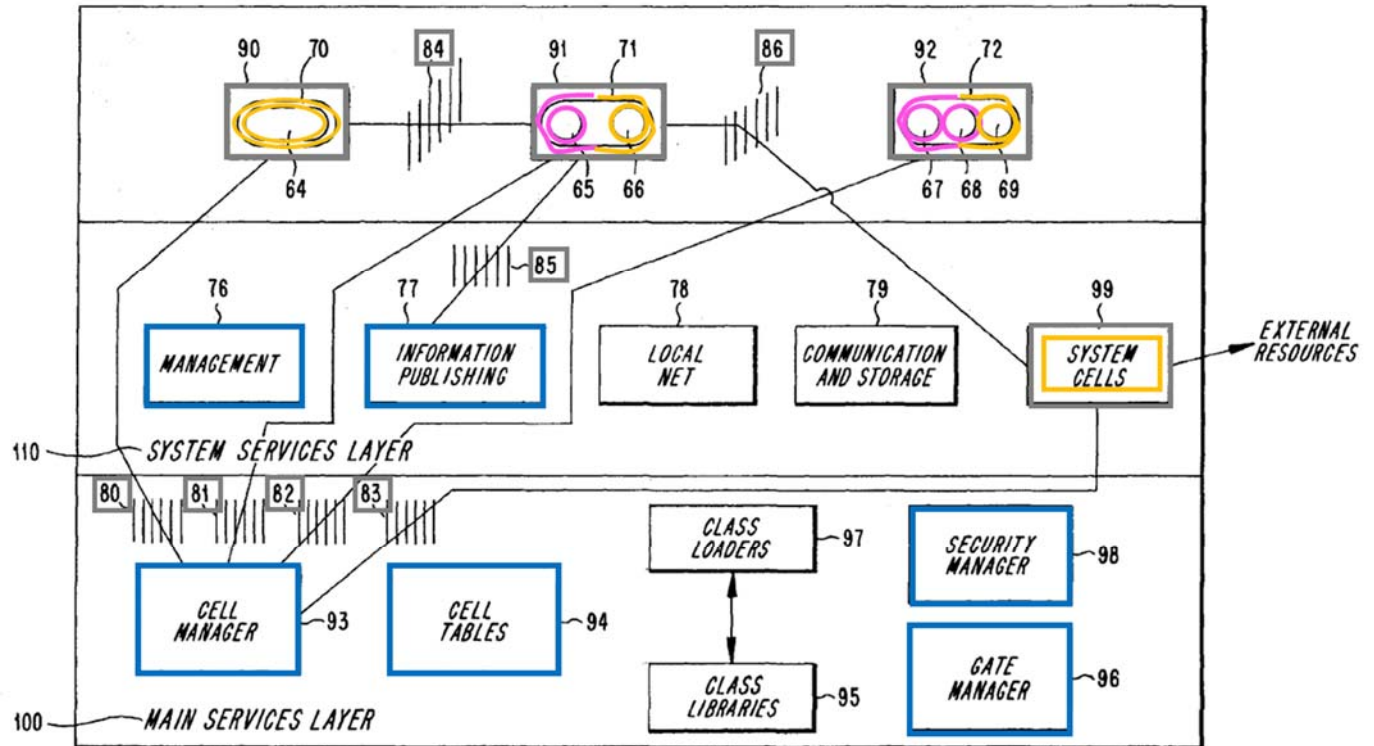


FIG. 6

EX1004, Annotated FIG. 6, 12:11-29, 20:23-31, 22:34-37, 20:10-22:30, 15:58-17:39, 4:44-50, 5:16-26, 9:20-40; EX1016, 4, 6-9, FIG. 4; EX1003, ¶¶325-326.

The management software includes “diagnostic” software, running on network operator server 24 (and/or service provider equipment 34) to “monitor”, “maintain and control” distributed service applications running on a service platform server(s) and one or more network operator and/or service provider server(s) in the service gateway system. EX1004, 9:20-41, 6:8-20, 3:2-5, 11:40-46; EX1016, 6, 8-9; EX1003, ¶327.

Vasell describes several additional operations for the service gateway system that are facilitated through these remote management services (e.g., undisturbed

services during software upgrades, dynamic resource allocation, etc.). EX1004, 5:16-26, 22:19-30, 22:6-18, 12:11-29, 2:56-59, 12:54-63, 20:45-49, 14:15-1; EX1016, 4, 6; EX1003, ¶¶329-331.

2. Alves

Alves (EX1008) provides an “*in-depth*” description of various technical implementation details of the OSGi open specifications (which are implemented in a Java run-time environment). EX1008, xiii-xiv; EX1003, ¶336.

Alves states “the OSGi platform is an ideal platform for cloud computing.” EX1008, 250, 266-269, xiii-xiv, 261-263. Alves identifies that OSGi’s standardized Enterprise implementation (EX1028) included communications between a **locally-installed** “**bundle**” that’s exporting a remote service in one OSGi framework instance and a [**remotely-installed**] **bundle** that’s importing the service in another [OSGi] framework instance” (the claimed “distributed **application**” with components installed on each **device**) using well-known protocols for providing **secure** communications between **software** components of a distributed **application** such as RMI, SOAP and REST. EX1008, xiv, 249-250, 252-255, 261-263, Figures 10.9-10.10; *see also* §V.F.3.a (identifying well-known secure implementations of these well-known protocols); EX1003, ¶¶337-338.

Alves discloses that, in a private/community/hybrid cloud OSGi Enterprise implementation, where the network administrator (rather than a cloud provider)

continues to “manage OSGi framework instances and their bundles” (e.g., EX1008, FIGS. 10.10, 10.12), a “cloud provider could give [the administrator] access to an OSGi bundle repository (OBR)” in the provider’s cloud, from which the administrator “could install, update, and uninstall bundles as needed”, and for which developers “can design and validate [their] applications using any OSGi implementation in a standalone environment before moving to the cloud.”

EX1008, 263, 266; EX1003, ¶¶339-340, 344. Alves discloses the importance of software developers “validat[ing]” (testing and certifying as successfully tested) their applications, e.g., before “moving [them] to the cloud” OBR, and technical implementation details for “effectively” and “eas[ily]” doing such application testing. EX1008, 126-130, 266; EX1003, ¶¶341-343.

Alves also discloses that, in a public cloud OSGi Enterprise implementation, where a cloud provider (and not the network administrator) “manages OSGi framework instances and their bundles” that are “hosted in machines managed by the cloud provider” (e.g., EX1008, FIGS. 10.11, 10.13), well-known “elasticity in the cloud” (what the Patent calls “cloud breathing”) is supported by the OSGi platform. EX1008, 266-268. EX1003, ¶345.

3. Rellermeyer

Rellermeyer discloses a “service... call[ed] CLOUDDEP... designed to turn OSGi services”—“objects that are specifically shared between [application]

components” called “*bundles*”—into elastic units of deployment which can be migrated and replicated across multiple **cloud** nodes to increase the availability and reliability of the administered service.” EX1011, 2-4; EX1003, ¶358. The **CLOUDDEP** service “can be architected using features **already provided** by OSGi” because, for example, (i) “the compositional approach of modular software... is key to building scalable and dependable systems across a varying set of **machines in the cloud**”, (ii) OSGi’s use of “services... as a design element... result[s]... [in] less entangled and more flexible” systems, and (iii) the “OSGi Enterprise Specifications” already detail “the **API** of the remote service **admin**” (“the **interfaces through which a possible topology manager can interact**”) and “the mechanism for importing **remote** services into... a **local** OSGI framework and exporting **local** OSGi services” to the **remote** service by creating “a **local** proxy for the **remote** service” and “binding [it] to the **remote** service”, which is “reachable through... a known URI or other identifier.” EX1011, 2-4; EX1003, ¶359.

“**CLOUDDEP**... takes over the role of a... local topology manager on each **node** of the cloud deployment” and “**monitors** the [**remote**] service invocations” on **local** (“**client**”) **nodes**, and explains “load-balancing” (Figure 2 (below, annotated), “case 1”) and “elastic scalability” (*id.*, “case 3”) usage cases, which provide the identical “cloud breathing” functionality disclosed in the ’823 Patent:

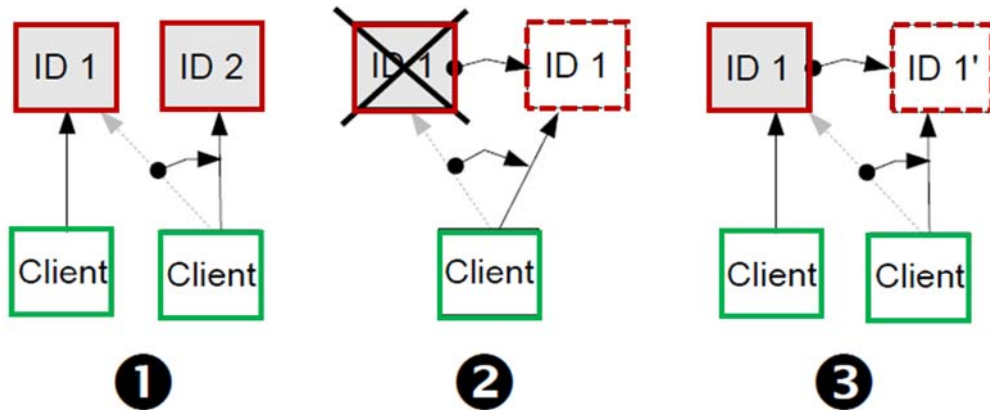


Figure 2. Different Use Cases for a Dependability Service. 1: Load balancing, 2: Failover, 3: Elastic Scalability.

EX1011, 4-5; EX1003, ¶360.

In Case 1 for “load balancing”, “**CLOUDDEP**... instrument[s] the **local** proxy to gather information about the frequencies of [**remote**] service invocation and response times”, and “[i]n the event that one **remote** service **instance** [“ID1”] **becomes overloaded**, detectable by an increase of invocations and an increase of the response time, **CLOUDDEP** can re-bind a **client** to a different [**remote**] service **instance** [“ID2”], thereby re-distributing the load among the [**remote**] service **instances.**” EX1011, 4-5; EX1003, ¶361. In Case 3 for “elastic scalability”, “[i]n the event of a [**remote**] service **over-utilization**, **CLOUDDEP** can start a new **instance** of the [**remote**] service [“ID1’”] and bind some of the **clients** to the new service” and “[s]cal[es] back to a smaller number of [**remote**] service **instances.**... by... moving all **clients** of a [**remote**] service to a different existing **instance**

[“ID1”] and then dropping the service **instance** [“ID1”].” EX1011, 5; EX1003, ¶362.

4. Hall

Hall (EX1009) “thoroughly describe[s] the important aspects of OSGi and show[s] how to use them” in the standardized Java run-time environment.

EX1009, xx; EX1003, ¶347.

Hall identifies “the OSGi framework is designed to take advantage” of “the built-in support for secure sandboxes” in “the Java platform”, including assigning permissions to “**bundles**”—called “boxlets within cells” or distributed **service application** “portions” in Vasell—, and having the AccessController and SecurityManager components of the JVM allow or deny access to **bundles** based on such permissions. EX1009, 438, 440-448, 464; *see also* §V.F.1; EX1003, ¶348.

Hall explains that another well-known Java security feature that “can be used for OSGi”, especially in environments where **bundles** (and associated upgrades) are developed by third parties, is digital signatures. EX1009, 457-463, 331-334, 471-476; *see also* §V.F.2; EX1003, ¶¶349-351. Hall states, “[i]n OSGi, the signer of a **bundle** is associated with it” and, “[w]ith this association,” the **management agent** of the OSGi environment would “grant permissions to a **bundle** based on its signers,” *i.e.*, based on the identity of the **bundle** developer. *Id.*; EX1009, 458-459, 476. Hall explains that this standardized “digital signing”, and

authenticity and integrity verification, was applicable to both original, and updated, **bundle** versions to render them “tamperproof.” *Id.*; EX1009, 331-333, 179; EX1003, ¶¶352-353.

Hall explains that the “JVM security manager”, of the JVM(s) (each of which, in the OSGi environment, is running on a host OS of a network device in which the **bundles** are respectively deployed), performs digital signature matching to determine these **bundle**-specific “permissions”, “control [each **bundle**’s] access to sensitive framework operations and information”, “control[] which services a **bundle** can provide or use”, make **bundle**-specific “deny-access decisions”, determine the local resources “required by the **bundle**”, “enforce” local resource restrictions “at execution time”, etc. EX1009, 471, 439-448; EX1003, ¶¶354-355.

5. The Motivation to Combine

As detailed in State of the Art §V.F.3.a, Vasell’s architecture and methods are foundational concepts (including implementation in a Java run-time environment) that were eventually standardized under OSGi. Technical implementation details that were intended to implement Vasell’s teachings are described in each of the published references cited in Grounds 1 and 2—Alves, Rellermeyer and Hall. Accordingly, the motivation for POSITAs to modify the teachings of Vasell, to include the features described in the published references for their intended purpose and to achieve predictable benefits with a reasonable

expectation of success is demonstrated by the references themselves. EX1003, ¶¶375-376.

For example, Vasell discloses a **service gateway system** in which connectivity-based services are implemented using **service applications** whose processing is distributed across respective Java software components, respectively executing in JVMs hosted on OSs of **local** and **remote (cloud)** servers and securely communicating with each other, where a network operator administrator remotely, via a **management interface**, and **management software modules** also executing on the **service gateway system servers**, **manages** the downloading, installing, upgrading and termination of these distributed **service application** components, and monitors and controls their implementation on the **service gateway system servers** (including dynamic device resource allocation), and for which many different service providers and third parties respectively develop the distributed **service applications**, and their underlying Java software components, for many different connectivity-based services. §VII.A.1; EX1003, ¶377.

POSITAs would be motivated to modify Vasell's teachings with a **centralized application repository** for storing distributed **applications**, for which developers can design and validate their **applications** using any OSGi implementation in a standalone environment before moving to the cloud, as taught by Alves because it provides the benefit of increased flexibility and security.

EX1008, 263, 266. Specifically, Alves describes technical details for (i) implementing OSGi-compatible **distributed systems** in various well-known cloud environments (including utilizing a cloud provider's **OBR** to store “validate[d]” (tested and certified as successfully tested) **bundles**), and (ii) using well-known protocols to provide secure communications between **locally-** and **remotely-** installed software components of distributed **applications** implementing services. §VII.A.2; EX1003, ¶¶378-379.

POSITAs would be motivated to modify Vasell's teachings with **management software** that provides for load balancing, failover and elastic scalability as taught by Rellermeyer. Specifically, Rellermeyer discloses a **management service** that, “using features already provided by OSGi” specifications, has the OSGi **management software** monitoring and controlling automatic load (resource usage) expansion/contraction across multiple **cloud servers**. §VII.A.3; EX1003, ¶380.

POSITAs would be motivated to modify Vasell's teaching with **management software** that requires digital signatures for distributed applications as taught by Hall to provide increased security for using third party applications. Specifically, Hall discloses technical implementation details for having the OSGi **management software**, and each “JVM security manager” of the JVMs (each respectively running on a network device's host OS in which the **bundles** are respectively

deployed), implement well-known, standardized digital signature, certificate and key exchange techniques to verify the respective authenticity and integrity of each downloaded **software bundle** (including updated versions). §VII.A.4; EX1003, ¶381.

POSITAs understood that the beneficial aspects that Alves, Rellermeyer and Hall (§VII.A.2-§VII.A.4) described for OSGi **distributed systems**—increased scalability, having a centralized **repository** for storing distributed **software application** components (and associated updates) developed by different application developers (of varying trustworthiness), increased security, being able to grant **application**-specific-permissions based on the respective **application** developer’s identity, and being able to elastically deploy cloud resources to increase the availability and reliability of services—would be identically beneficial to Vasell’s foundational **service gateway system**. EX1003, ¶382; *see also* State of the Art §V.F.1-§V.F.3.b.

POSITAs would be motivated to modify Vasell’s teachings to utilize a **cloud-provider-application repository** of “validate[d]” **applications** in a cloud environment, implement inter-application communication protocols with well-known secure communication capabilities, implement well-known, standardized digital signature techniques, and to have its remote **manager** monitor and control

load expansion/contraction across multiple **cloud servers**, in order to achieve the known and predictable benefits associated therewith. *Id.*; EX1003, ¶383.

In view of this section (and discussed further below in the claims analysis), Petitioner has demonstrated at least:

- i. Vasell and Alves themselves provided the motivation that would have led POSITAs to implement Vasell’s network of **service provider servers** and **network operator server(s)**, as a “cloud” using a **cloud-provider-application repository**, to implement “many different types of services” using “validate[d]” software components of distributed **applications** developed in an “open” environment including “different service providers”, “third party developers”, and “third-party markets for... services.” EX1003, ¶385.
- ii. The general knowledge in the art was that Enterprise OSGi’s and Vasell’s shared features—*e.g.*, modular software applications; **distributed** execution of **service applications** in a two-tier, multiple machine (**local** and **remote (cloud) servers**), virtualized (JVM) architecture; dynamic **service application** component updates without affecting the concurrent execution of **service applications** on the same **servers**; and dynamic resource usage (load) monitoring and control—made OSGi (and Vasell) an “ideal” system to implement in a well-

known “cloud” environment, where a private/community/hybrid cloud implementation would continue to be managed by the network operator as disclosed by Vasell, and to utilize a cloud-provider-application-store to store “validate[d]” distributed service application components for respective installation in Vasell’s service gateway system servers. EX1003, ¶¶386-390.

- iii. The beneficial aspects of having a centralized repository/store for software developed by different application developers, and requiring each developer to “design and validate” its software in its own “standalone environment before moving” such software to the cloud-provider-application-store, as described in Alves, would be accomplished as a mere application of a known centralized repository/store solution to a known distributed system ready for improvement to achieve known and predictable results. EX1003 ¶391.
- iv. Vasell and Rellermeyer themselves provided the motivation that would have led POSITAs to improve, “using features already provided by OSGi” specifications, Vasell’s remote manager monitoring and dynamically allocating service gateway system resources using Rellermeyer’s teachings of the OSGi management

software monitoring and controlling load expansion/contraction across multiple cloud servers. EX1003, ¶392.

- v. The beneficial aspects of this combination, including turning Vasell’s connectivity-based services “into elastic units of deployment which can be migrated and replicated across multiple cloud nodes to increase the availability and reliability of the administered service”, would be accomplished “using features already provided by OSGi” specifications, and with the knowledge that OSGi’s and Vasell’s shared features made OSGi (and Vasell) an “ideal” system to implement in a well-known “cloud” environment, would give POSITAs confidence that the combination would have a high likelihood of success. EX-1003 ¶393.
- vi. Vasell and Hall themselves provided the motivation that would have led POSITAs to implement Vasell’s administrator, and management software, managed upgrades of Java software components of distributed service applications, in Vasell’s “open” development environment, using, as described in Hall, well-known, standardized digital signature techniques in OSGi’s JVM environment (shared with Vasell) to respectively verify the authenticity and integrity of each such software component. EX1003, ¶394.

vii. The beneficial aspects of this combination, including rendering each distributed application “portion” (and updates) “tamperproof”, authenticating the provider of such updates (by verifying that the signer has access to the unique private key), ensuring that the content of such updates is unmodified, and allowing Vasell’s management services to grant Vasell’s application-cell-specific-access-levels to each such distributed application “portion” (and updates) based on the respective developer’s identity (signature), would be accomplished using well-known and standardized security techniques in OSGi’s and Vasell’s shared centralized management and virtualized (JVM) architecture, as detailed by Hall, and, accordingly, POSITAs would have been confident that the combination would have a high likelihood of success. EX-1003 ¶395.

B. Ground 1: Detailed Application of Vasell in view of Alves and Rellermeyer to Claims 1-2, 12-15 and 19

Claim 1

[1.0] A system comprising:

To the extent the preamble is limiting, Vasell discloses this limitation because it discloses a “service gateway system” (also called “e-service infrastructure”) that utilizes “distributed processing” across “servers or processors” connected to each other over “an Internet Protocol (IP) network such as the

Internet” (system). The **service gateway system** is shown in Vasell’s Annotated Figure 2 (below), which includes hardware: (1) a “**service platform server 22**” “connected to the local area network 10” and connected to (2) a “**network operator server 24** via the Internet”, which is connected to (3) a plurality of **service provider equipment 34** “via... Internet 26”:

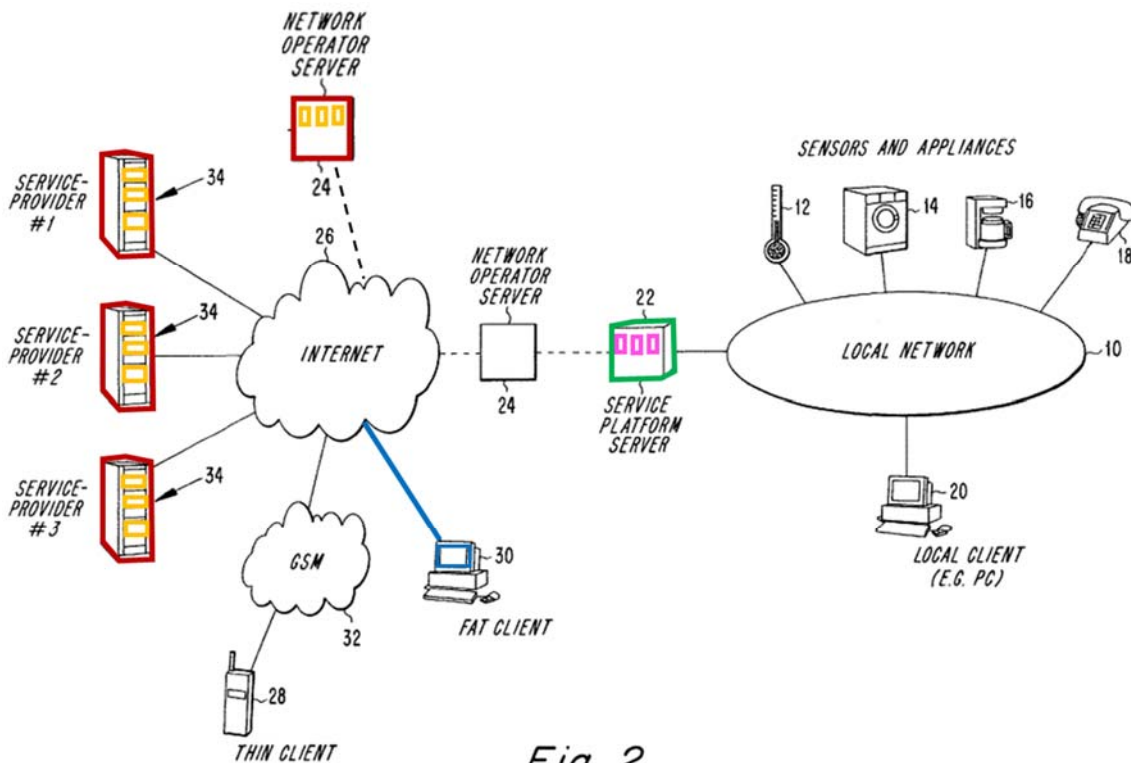


Fig. 2

EX1004, 4:31-47, 5:61-6:20, 6:62-7:6, 7:12-18; EX1016, 4, 6-9, FIGS. 2, 5; EX1003, ¶¶397-398.

Vasell discloses that the **distributed** “system software”, of each of the **local** and **remote** servers of the **service gateway system**, includes “a Java runtime in the form of a Java virtual machine (JVM)” and an OS hosting the JVM, which could

be “[a]ny 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.” EX1016, 4, 6-9, FIGs. 2, 5; EX1004, 18:27-30; EX1003, ¶¶401-402.

Vasell discloses the service gateway system “provides a platform for connectivity based services”, where each of a plurality of distributed software applications—called “service applications”—implementing a respective “service” for end users. Each “service application may be distributed among various pieces of equipment which are geographically separated”, such that each distributed service application has a component installed on a service platform server and a component installed on a network operator server and/or service provider equipment. EX1004, 4:31-50, 2:64-3:6, 1:41-45, 8:53-9:18, 12:21-33, 11:11-58; EX1004, 4, 7-9; EX1003, ¶¶399-400.

Accordingly, the above-annotations in Annotated FIG. 2 show a plurality of service applications (each including a plurality of distributed software components). Specifically, each component of a service application is respectively hosted on a JVM running on the respective OS of the service platform server 22, the network operator server(s) 24, and the service provider equipment 34. *Id.*; EX1004, 4:31-47, 5:56-6:20, 6:62-7:14; EX1016, 7-9; EX1003, ¶403.

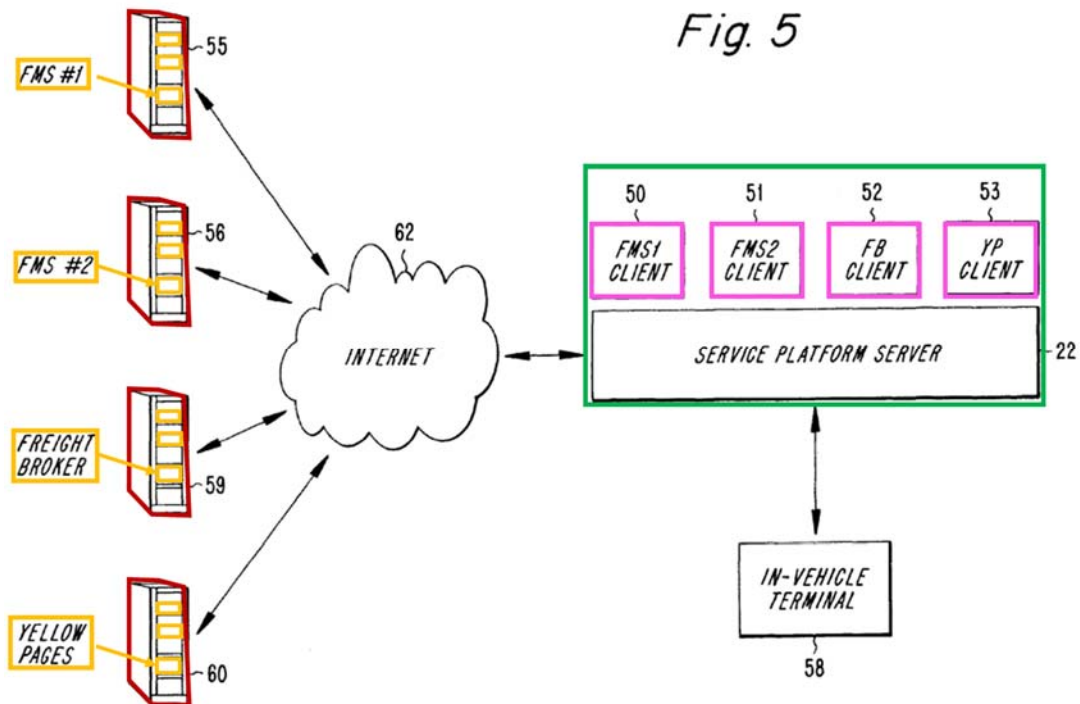
[1.1] a programmable network device adapted to host a plurality of network device applications;

Vasell discloses this limitation. Vasell discloses “service platform server 22” [*programmable network device*] is “an edge server” and “the hardware portion of a service gateway or services platform” (Annotated FIG. 2), and states it “may provide service support for” “household users and residences..., an entire office building, an industrial plant, or even a campus-type organizational facility.” EX1004, 5:61-6:20, 6:45-49, 10:32-41; EX1003, ¶¶405-406.

Vasell discloses “service platform server 22 may contain... part of... software... applications... used to implement one or more connectivity based services which have been requested and/or subscribed to by the end user” [*adapted to host a plurality of network device applications*]. EX1004, 6:21-25, Annotated FIG. 2; EX1003, ¶407.

Vasell discloses several examples of “distributed” service applications in the service gateway system, and of “portions”/components thereof hosted on service platform server 22. *Id.*; EX1004, 9:42-61, 8:53-9:18. For example, Annotated FIG. 5 illustrates “client software packages 50-53 which are running at any given time on the service platform server 22”:

Fig. 5



EX1004, 11:11-39, 13:28-36; EX1003, ¶¶408-412.

[1.2] a programmable cloud device adapted to host a plurality of cloud applications,

Vasell alone, in view of Alves, or further in view of Rellermeyer discloses this limitation.

Vasell discloses “service provider equipment 34, which may include computer systems or servers, may be the initial repository for software used to implement connectivity based services” [programmable cloud device]. EX1004, 7:4-20, 11:40-46, 12:31-33; EX1003, ¶414.

Vasell also discloses “network operator server 24 equipment may include one or more servers, i.e., relatively powerful computers operating in conjunction

with one or more secondary storage devices” [another *programmable cloud device*], “handle[d] control for the operation, maintenance and support of the service[s] implementation.” Vasell further discloses “network operators using the **network operator servers 24** may act as intermediaries between service providers and the end users of those services.” EX1004, 6:45-7:3, 6:16-20, 9:25-40, 11:40-46; EX1016, 5-9; EX1003, ¶¶405-417.

Moreover, Vasell discloses the **service provider equipment** of each service provider, and **network operator server** of each network operator, may each host respective “**portions**”/components of “distributed” **service applications** to implement each of a respective plurality of services [*programmable cloud device hosting a plurality of **second network applications***]. EX1004, 8:53-9:24, Annotated FIG. 2, 2:64-66, 11:12-46, Annotated FIG. 5, 13:28-36, 11:47-58 (example of plural services provided by same **service provider**: (i) “automated meter reading”, (ii) “real-time pricing”, (iii) “automated load control”), 12:27-29; EX1016, 4-5, 7-9; EX1003, ¶¶419-422.

Accordingly, POSITAs understood, as explained in the State of the Art §V.F.3.b, Vasell’s disclosed network of **network operator server(s)** and **service provider servers** accessed “transparent[ly]” through the Internet, each respectively hosting **service application** “portions”/components on a JVM running on the OS on the **server’s** hardware to implement services through the distributed **service**

gateway system, to constitute a “*cloud*”, and each such **server** a *cloud device*.

EX1004, 2:60-66; §V.F.3.b (EX1033, 94; EX1052, xxv, 3, 45; EX1030, [0038]; EX1057, 153); EX1003, ¶418.

To the extent PO asserts that Vasell does not explicitly disclose its **service provider servers**, and **network operator server(s)**, as “*cloud*” devices, as discussed in the Motivation to Combine §VII.A.5, it would have been obvious to POSITAs to implement these servers as a “cloud” as a mere design choice with a reasonable expectation of achieving well-known benefits. EX1003, ¶423; EX1004, 7:12-14.

As discussed in the Motivation to Combine §VII.A.5, and as established by Alves, POSITAs would be motivated to select the known design choice of implementing Vasell’s **service provider servers**, and **network operator server(s)**, in, for example, a private/community/hybrid cloud as taught by Alves, where the **service gateway system** would continue to be **managed** by the network operator as Vasell discloses, with a reasonable expectation of achieving known and predictable benefits such as rendering “[t]he **system**... more scalable, because... we have a flexible setup, where we understand which **components** can be replicated and how best to partition the requests” because it was known to POSITAs that Vasell’s disclosed features (which it shared with OSGi) made Vasell’s **service gateway system** an “ideal” system to implement in a cloud environment. §VII.A.5

(EX1008, xiii-xiv, 250, 261-263, 266-268, FIGS. 10.10-10.13); EX1003, ¶¶425-427.

Furthermore, to the extent PO asserts that, (i) in order to be a programmable “cloud” device, Vasell’s **remote management** functionality must include monitoring and control functionality for load (resource usage) expansion and contraction across Vasell’s **service provider servers** and **network operator server(s)**, and (ii) Vasell in view of Alves does not disclose *how* this is explicitly accomplished, for the reasons discussed in the Motivation to Combine section, POSITAs would be motivated to modify Vasell’s teachings with Rellermeyer’s teachings of providing load balancing and elastic scalability “using features already provided by OSGi” specifications. §VII.A.5 (EX1008, 266-268; EX1011, 2-5); EX1016, 9; EX1003, ¶¶428-430.

[1.3] wherein the plurality of network device applications and the plurality of cloud applications are in secure communication with each other to form distributed applications; and

Vasell alone, or in view of Alves, discloses this limitation. Vasell discloses **distributed service applications**, implementing connectivity-based services in the **service gateway system**. EX1003, ¶¶431-432.

As an example of **distributed service applications** in the **service gateway system**, Vasell discloses, referencing Annotated FIG. 2, that “a **first portion** of **software** associated with an environmental service may reside... on the **#1 service**

provider equipment... A second portion of the software used to implement [this service]... may reside as part of the network operator server... Likewise, a third portion of the software used to implement this service may reside on the service platform server.” [the plurality of network device applications and the plurality of cloud applications... form distributed applications]. EX1004, 8:53-9:24, 2:64-3:1, 24:49-51, 11:47-58 (example of plural services provided by same service provider: (i) “automated meter reading”, (ii) “real-time pricing”, (iii) “automated load control”), 12:27-29; EX1016, 4, 7-9; EX1003, ¶¶433-435.

In other words, Vasell discloses a plurality of “distributed” software applications—called “service applications”—each implementing a “service” for end users, and each having (i) a component (“portion”) installed on a service platform server, and (ii) a component (“portion”) installed on a network operator server and/or service provider equipment. *Id.*

Another example of “distributed” service applications in the service gateway system, where each distributed application has a component at a client location and a component at a service provider location, is in Vasell’s Annotated FIG. 5. Vasell discloses, using a trucking industry example, mobile service platform server (“outfitted” on a truck) (e.g., in a JVM running on the server OS) has various “service software client packages” (50-53) loaded thereon that communicate with associated software components loaded on service provider servers (55, 56, 59, 60)

(e.g., in respective JVMs running on each **server** OS) to provide services to trucking industry users. EX1004, 11:13-46; EX1016, 4, 9; EX1003, ¶¶436-439.

POSITAs understood that Vasell also discloses that the underlying **software modules** that collectively form each of these examples of **distributed service applications** *are in secure communication with each other* for several reasons. EX1003, ¶440.

First, per [1.0], Vasell discloses that each component (“portion”) of a distributed **service application** is respectively hosted on a “Java Virtual Machine (JVM)” running on the respective OS of the **service platform server**, the **network operator server(s)**, and the **service provider equipment**, in the **service gateway system**. EX1016, 4, 7-9; EX1004, 18:27-30; EX1003, ¶¶441-442.

Second, Vasell emphasizes: “Applicants consider **secure** implementation of these connectivity based services to be important” and, “[s]ince various service providers, network providers and, possibly, end users, will share infrastructure, it is important that information and **communication associated with each implemented connectivity based service**”—*i.e.*, communications between the respective software components that collectively form each of these “distributed” **service applications** to implement each such service—“**be secure.**” EX1004, 11:59-64; EX1003, ¶¶443-444. Specifically, Vasell expressly envisions “**widespread usage of... security techniques including data encryption... so**

that the implementation, monitoring... associated with these services **is robustly defended against intrusion and manipulation.**” *Id.*; EX1016, 3, 6.

Third, consistent with the well-known sandboxing of the run-time environment of JVMs running on the host OSs of **service gateway system servers**, Vasell discloses that each of the underlying software components (each a “boxlet” within a “cell”) that collectively form each of the distributed **service applications** is “designated into various categories having different degrees of access restriction”, where “access rules” (**administrator-defined** for each application’s cell and for application cell “groups”) are **controlled** and **enforced** using, in part, “inter-cell communication... controlled interfaces” called “gates” (**managed** by “**gate manager**”) for **application component-to-component** communications. *Id.*; EX1004, 11:64-12:10, 4:58-65, 13:28-36; EX1016, 8, Figure 4; EX1003, ¶445.

Accordingly, although Vasell uses different words than the ’095 Patent, POSITAs understood Vasell discloses its distributed **service applications** being formed by the respective underlying software components securely and “transparent[ly]” communicating with each other using well-known “data encryption” protocols, managed “gates” enforcing cell-specific “access rules”, and Java APIs from the respective virtualized, sandboxed, and “higher level of abstraction” environment of respective JVMs running on the respective OSs of the **service platform server** or **remote (cloud) server** on which they are hosted [*in*

secure communication with each other to form a distributed application] with the same, or an equivalent, structure as the '823 Patent. *Id.*, see §V.C; EX1003, ¶¶446-448.

To the extent PO asserts that Vasell does not explicitly disclose **secure** communications between the underlying software components of Vasell's distributed **service applications**, POSITAs would have been motivated to **securely** implement Vasell's inter-application communications as discussed in the Motivation to Combine section. Indeed, as discussed in the Motivation to Combine section, POSITAs would have been motivated to use well-known protocols (*e.g.*, RMI, SOAP, REST) as taught by Alves, which POSITAs understood, as explained in the State of the Art section (§V.F.3.a), would provide secure communications between Vasell's Java software components of a distributed **service application**, respectively running in a virtualized (JVM) environment (on a host OS of a **service gateway system server**), and with a reasonable expectation of achieving the increased security benefits expressly disclosed by Vasell. *Id.*; §VII.A.5 (EX1008, 249, 252-255, 261-262, Figure 10.9); §V.F.3.a (identifying well-known secure implementations of RMI, SOAP, REST); EX1003, ¶¶449-452.

[1.4] wherein the plurality of network device applications and plurality of cloud applications device form unified capabilities enabling a plurality of upper layer application programming

interfaces (APIs) to program the plurality of network device applications and plurality of cloud applications independent of network device hardware and cloud device hardware.

Vasell discloses this limitation.

Vasell discloses that hardware-independence of the *application programming interfaces (APIs) to program the service applications* is important due to the market-driven “open” nature of the *service gateway system* (also called “*e-service infrastructure*”). See [1.0]; EX1003, ¶¶454-455.

For example, Vasell discloses “a new *e-service infrastructure* is required— an **open** platform that can be used by **several independent** service providers.” EX1016, 9; *see also, id.*, 3 (“[t]he market seems to favor an **open e-service platform**...the strength of the network operator depends on the number of service providers using the network, and an operator will therefore try to keep the **platform** as **open** as possible.”). Additionally, for example, Vasell discloses “[t]he *service gateway system* according to the present invention facilitates the development... of services... [by] **different services providers**” where “[e]ach service... [is] implemented as software *service applications*.” EX1004, 2:60-66; *see also, id.*, 3:10-19, 5:30-33, 10:18-23, 2:60-66, 13:37-43; EX1003, ¶¶456-457.

Given the “open” nature of the *service gateway system*, Vasell discloses, for example, that “[t]he development environment [for *application software*] **must** follow the ‘**write-once, run-everywhere**’ **maxim** and should be **based on Java**

standards.” EX1016, 4; *see also* [1.0]-[1.3] (describing Vasell’s disclosure that, in the **service gateway system**, each **service application** “portion”/component is respectively hosted on a JVM running on the respective OS of the **service platform server**, the **network operator server(s)**, and the **service provider equipment**); EX1003, ¶458.

POSITAs understood that Vasell discloses that *the plurality of network device applications and plurality of cloud applications* (the respective underlying Java software components in secure communication with each other to form distributed Java **service applications**) *form unified capabilities enabling a plurality of upper layer application programming interfaces (APIs) to program the plurality of network device applications and plurality of cloud applications independent of network device hardware and cloud device hardware* for at least five reasons. *Id.*; EX1003, ¶¶460-462. Specifically, POSITAs understood that Vasell discloses this limitation given that (1) the distributed Java **service applications** are naturally modular; (2) the respective underlying Java software components, of such distributed **service applications**, use standardized communication protocols to communicate with each other; (3) standard Java APIs are used for **their** development; (4) “flexible”, “transparent”, and standards-compatible interfaces are used for **their lifecycle management** in the **service gateway system**; and (5) these standard development and **management** APIs are usable regardless of the

respective **service platform servers'** hardware and **remote (cloud) servers'** hardware on which the respective JVMs run on the respective OSs. EX1016, 4, 6, 8-9; EX1004, 2:60-3:1, 5:15-35, 22:35-37, 20:52-53, 16:59-60, FIG. 6; EX1003, ¶¶462-463.

Claim 2

The system of claim 1, wherein the programmable network device and the programmable cloud device each include an access network interface unit configured to send and receive communications and a processor with a memory associated with the access network interface unit.

Vasell discloses this limitation. Vasell discloses (1) “**service platform server**” [*programmable network device*] includes “access technology” to send and receive communications in the LAN (*e.g.*, “Ethernet cable,... modem or other gateway connection”) and over “Internet or other communication network” (*e.g.*, “wireless...connection”), and (2) “**network operator server**” and “**service provider equipment**” include “access technology” to communicate over, and be “connected to”, “an Internet protocol (IP) network such as the Internet 26 or other network technology... [*e.g.*,] wireless”) [*each of the programmable network device and programmable cloud device include an access network interface unit to send and receive communications*]; EX1003, ¶¶464-465.

Vasell discloses the “**cell manager**” monitors resource usage, including “CPU time, memory”, on each of these **service gateway system servers** by the

service applications, and identifies that the “distributed **processing** techniques... [are] implemented using **software** operating on two or more of the servers or **processors** associated with the architecture illustrated in FIG. 2” [*a processor with a memory associated with the network interface unit*]. EX1004, 20:45-49, 14:15-17, 20:55-58, 22:49-53, 8:53-58, 9:13-19; EX1003, ¶¶466-467.

Claim 12

The system of claim 1, further comprising a virtual fabric which provides a secure communication layer for said at least one of the plurality of network device applications and said at least one of the plurality of cloud applications.

Vasell alone, or in view of Alves, discloses this limitation. Per [1.3], Vasell alone or in view of Alves, discloses forming distributed **service applications** by their respective underlying Java software components being in **secure** communication with each other; EX1003, ¶¶468-469.

For at least the reasons explained in [1.3], although Vasell uses different words than the '823 Patent, POSITAs understood Vasell, alone or in view of Alves, discloses, , with the same, or an equivalent, structure as the '823 Patent (§V.C), that the **service gateway system servers** includes *a virtual fabric which provides a secure communication layer* over which the respective underlying Java software components (each a Java “boxlet” within a “cell”), of Vasell’s distributed **service applications**, securely and “transparent[ly]” communicate with each other using well-known “data encryption” protocols, managed “gates” enforcing cell-

specific “access rules”, and the respective virtualized, sandboxed, and “higher level of abstraction” environment of a JVM running on the respective OS of the **service gateway system server** on which they are hosted. EX1003, ¶¶470-475.

Claim 13

The system of claim 1, further comprising: a second programmable network device which includes at least one of the plurality of network device applications which is the same as at least one of the plurality of network device applications in the programmable network device and where each of the network device applications can communicate directly.

Vasell alone, or in view of Alves, discloses this limitation. Vasell discloses that “the **service gateway system** is accessed by and shared with multiple service providers and service gateway users”, and that the “service gateway network may potentially range from several hundred thousand to millions of service gateway units.” EX1004, 4:51-54, 5:16-20. POSITAs understood Vasell discloses, including based on the examples of “service providers” and examples of “connectivity based services” being provided to “various end users” (residences and/or businesses), different **service platform servers** running the same distributed” **application “portion”/component** thereon so that the same connectivity-based service is provided to different end users via the **service gateway system**. *Id.*; EX1004, 6:45-49, 3:52-57, 10:17-38, FIG. 3, 11:47-58; EX1003, ¶¶476-478.

Moreover, per [1.3], POSITAs would have understood Vasell, alone or in view of Alves, discloses that, in implementing **secure** communications between the respective underlying Java software components, which form the “distributed” **service applications** using well-known communication protocols (e.g., RMI, SOAP or REST), the Java **components/“portions”** hosted in the JVM running on the OS of the **service platform server** *can communicate directly* with the Java **components/“portions”** hosted in the JVM running on the OS of the **remote (cloud) server** (e.g., during a secure communication session). EX1003, ¶¶479-480.

Claim 14

The system of claim 1, further comprising: a distributed resource service (DRS) which is capable of providing at least one service from a group consisting of: exposing application programming interfaces (APIs) to other applications; configuring and managing platform resources; policy enforcement and authorization of the plurality of network device applications and the plurality of cloud applications to access platform resources; and policy conflict resolution.

Vasell discloses this limitation. Vasell discloses “[e]ach of the [application-]cells... may be characterized according to its **quota of resources** (e.g., CPU, memory, persistent storage....” EX1004, 14:15-17, 13:28-36, 14:2-7. Vasell also discloses a “**cell manager**” (itself a “**boxlet** ... executing within [a] cell”), managed via Vasell’s “**management system service**”, “handles **resource management** so that no single one of the [application-]cells... **consumes too much of the service**

gateway system... resources, e.g., CPU time, memory, persistent storage, and the like”, and where this service gateway system “resource management” is facilitated by “diagnostic software” to “monitor”, “maintain and control” the “distributed” service applications. [configuring and managing platform resources]. EX1004, 20:45-49, 20:23-31, 20:36-43, 9:20-41, 6:8-20, 8:53-9:18, 14:2-7, 12:21-29, Annotated FIG. 6 (below):

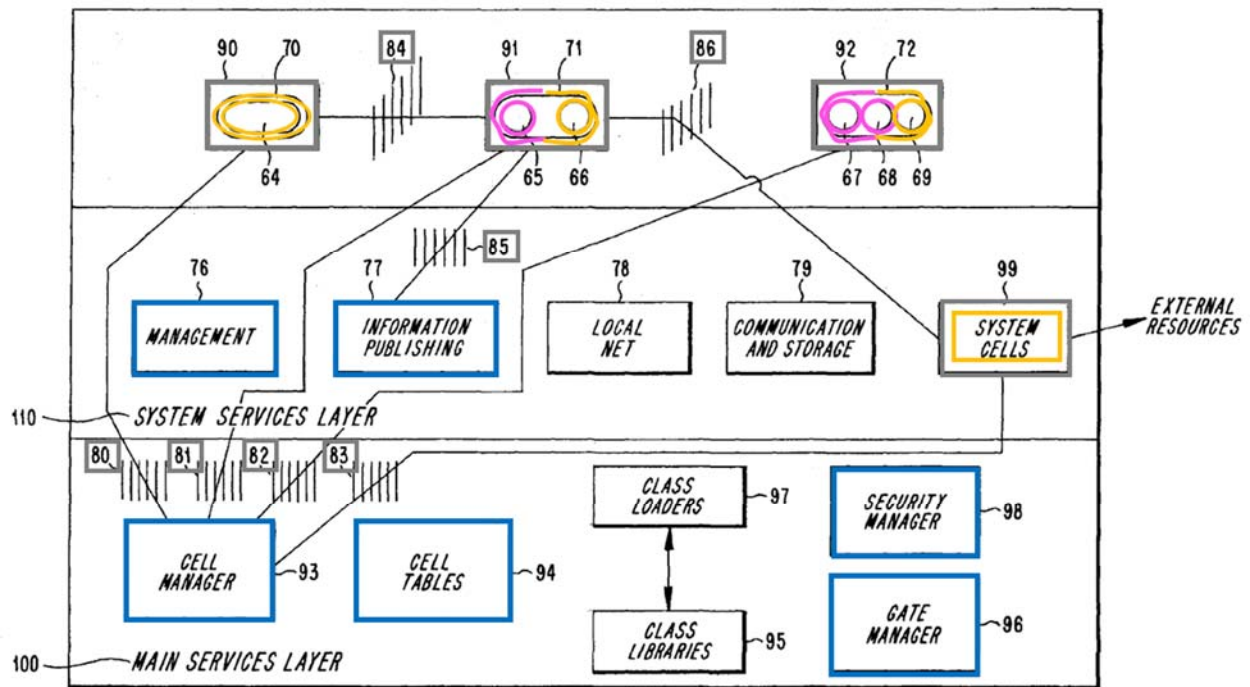


FIG. 6

EX1003, ¶¶481-482, 484.

Vasell further discloses that this “cell manager” functionality includes “resource requirements... be[ing] dynamically allocated based upon a predetermined scheme of maximum allowed or minimum required resources, for a given situation” so that “different ones of the service applications” do not

“inadvertently interact in such a way that would cause them to... detrimentally affect each other” and to “allow[] for resource tradeoffs between the **service applications**... so as to maximize user benefit for a given amount of resources.” *Id.*; EX1004, 12:51-63, 13:12-27; EX1003, ¶483.

Vasell further discloses controlled implementation of “listeners” in the **service gateway system**, using “proxies” to send generated “event notifications... upon the occurrence... of a specified event” between “distributed” **service application** components and/or between a **service application** and the “main” and/or “system” “**services layer**” components (*e.g.*, “**service boxlet[s]**”, “**diagnostic software**”). *Id.*; EX1004, 13:39-48, 18:16-19, 19:5-41, Annotated FIG. 6, 12:21-29; EX1003, ¶485.

Accordingly, POSITAs understood Vasell discloses that, as part of implementing these “resource **management**”, enforcement of [application-]cell-specific resource “quota[s]”, and “dynamic[] allocat[ion]” of “resource requirements” functions for the plurality of distributed **service applications** implementing services in the **service gateway system**, respective instructions would be provided from Vasell’s “**management system service**” to respective **management software** (*e.g.*, “**service boxlet[s]**”, “**diagnostic software**”) distributed between the **service gateway system servers** [*a distributed resource service (DRS)*]

which is capable of ... configuring and managing platform resources]. EX1003, ¶¶486-487.

Claim 15

The system of claim 1 further comprising: a distributed content service wherein contents generated by the at least one of the plurality of network device applications are shared with at least one of the plurality of cloud applications.

Vasell discloses this limitation. Specifically. Specifically, POSITAs understood that part of the secure communications between the distributed **service application** components (see [1.3]) would include generated content, such as, for implementation of a “freight management service”, “**client software package**... on the **service platform server 22** [may] provide **freight company FMS #1** with information regarding the current location of the truck and its anticipated arrival time of delivery” [*distributed content service wherein contents generated by the at least one of the plurality of first network applications may be shared with the at least one of a plurality of second network applications*]. EX1004, Annotated FIG. 5, 11:13-29, 8:53-9:3, 9:13-24, Annotated FIG. 2; 19:5-41 (“gates” and “proxies” used in the **service gateway system** to share “arguments and return values”, “nonserialized objects” and “input/output data streams” between respective **service-application-specific-cells**); EX1003, ¶¶488-491.

Claim 19

[19.0] A system comprising:

Vasell discloses this limitation per [1.0].

[19.1] a programmable network device adapted to host a plurality of network device applications;

Vasell discloses this limitation per [1.1].

[19.2] a programmable cloud device adapted to host a plurality of cloud applications

Vasell alone, in view of Alves, or in view of Alves and Rellermeier, discloses this limitation per [1.2].

[19.3] wherein the plurality of network device applications and the plurality of cloud applications are in secure communication with each other to form distributed applications;

Vasell alone, or in view of Alves, discloses this limitation per [1.3].

[19.4] wherein the plurality of network device applications and plurality of cloud applications device form unified capabilities enabling a plurality of upper layer application programming interfaces (APIs) to program the plurality of network device applications and plurality of cloud applications independent of network device hardware and cloud device hardware.

Vasell discloses this limitation per [1.4].

[19.5] an application management portal capable of managing life cycles of the plurality of network device applications and plurality of cloud applications; and

Vasell discloses this limitation. Vasell discloses the **service gateway system** is “**remotely managed**” via “**management system service**” of “system services layer 110”, “implemented in the form of the **boxlets**[**]** executing within cells”, and that

“provides **an external interface** through which the **service applications** ... may be **downloaded, installed, removed**, executed, and controlled”, which are management operations that Vasell details as being performed by a “**cell manager**” using a “**cell table**” and “facilitate **management** of the **service applications**... **lifetimes**” [*application management portal managing life cycles of the plurality of network device applications and plurality of cloud applications*]. EX1004, Annotated FIG. 6, 12:11-29, 20:23-34, 22:34-37, 20:49-53, 20:10-22:30, 15:58-17:39, 4:44-50, 5:16-26; EX1016, 7-8, FIGS. 4, 9; EX1003, ¶¶498-502.

[19.6] an infrastructure application marketplace in communication with the application management portal, said infrastructure application marketplace capable of providing tested distributed applications to the application management portal.

Vasell in view of Alves discloses this limitation. Vasell discloses **management services** that provide “an **external interface** through which **service applications** ... may be downloaded, installed...” [*application management portal*], and the **service application** development environment of the **service gateway system** as an “open” environment, including “**different** service providers”, “**third party** developers” and “nonspecialists”, and “open[ing] **third-party markets** for... services.” EX1004, 3:10-15, 5:27-33, 10:18-23, 13:37-43, 22:35-37; EX1016, 4, 7-9; EX1003, ¶¶503-506.

Although Vasell also identifies that “**service provider equipment**... **may** be the initial repository for **software** used to implement connectivity based services” of such service provider, and that “**network operator server**... may need to **receive some portions** of the **service application software** from the service provider” to then “**download some or all of that [received] software** to the **service platform server**” [*distributed applications to the application management portal*], Vasell does not explicitly disclose a **centralized application repository** storing the respective **service applications** developed by each of the “**different** service providers”, “**third party** developers” and “nonspecialists” in Vasell’s intended “open” development environment. *Id.*; EX1004, 7:14-18, 8:29-36, 8:47-9:18; EX1003, ¶507.

As discussed in the Motivation to Combine section, POSITAs would have been motivated to modify Vasell’s teachings with Alves’ teachings of having a cloud provider give Vasell’s **service gateway system management services** access to its **application store**, as were well-known in the art (*see* State of the Art §V.F.3.a-§V.F.3.b), in which “validate[d]” (already-tested and certified as successfully tested) **service applications** would be stored, and from which Vasell’s **management services** could, as needed, download such **applications** for installation in the **service gateway system servers** [*infrastructure application marketplace in communication with the application management portal and capable of providing*

tested distributed applications to the application management portal] as mere application of a known solution to a known **distributed system** ready for improvement to achieve known and predictable results. §VII.A.5 (EX1008, 263, 266-267, FIG. 10.12, 126-130); EX1003, ¶¶508-513.

C. Ground 2: Detailed Application of Vasell in view of Alves, Rellermeyer and Hall to Claims 3-5, 7-8 and 18

Claim 3

[3.0] The system of claim 1 further comprising:

Vasell discloses this limitation per [1.0].

[3.1] an application management portal capable of managing life cycles of the distributed applications; and

Vasell in view of Alves discloses per [19.5].

[3.2] an infrastructure application marketplace in communication with the application management portal, said infrastructure marketplace capable of providing tested and certified distributed applications to the application management portal.

Vasell in view of Alves, or Vasell in view of Alves and Hall, discloses this limitation. Per [19.6], POSITAs would have been motivated to modify Vasell's teachings with Alves' teachings of having a cloud provider give Vasell's **service gateway system management services** access to its **application store**, as were well-known in the art (*see* State of the Art §V.F.3.a-§V.F.3.b), in which "validate[d]" (already-tested and certified as successfully tested) **service applications** would be

stored, and from which Vasell's management services could, as needed, download such applications for installation in the service gateway system servers [infrastructure application marketplace in communication with the application management portal capable of providing tested and certified distributed applications to the application management portal]. EX1003, ¶¶516-520.

To the extent PO asserts that Vasell in view of Alves does not disclose “certified” applications in the cloud-provider-application-store, as discussed in the Motivation to Combine section, POSITAs would have been motivated to modify the teachings of Vasell in view of Alves with Hall's teachings of requiring that OSGi-application-developers use well-known digital signature, key exchange and certificate techniques to “digitally sign” their respectively developed distributed Java application modules, whereby a “well-known (trusted)” party certifies a respective application's authenticity if it is digitally signed by the respective unique private key issued to the accompanying certificate-identified developer. *Id.*; §VII.A.5 (citing EX1009, 179, 331-334, 457-463, 472-476); EX1003, ¶¶521-525.

Claim 4

The system of claim 3, wherein the application management portal is capable of at least one of the group consisting of: receiving new applications from the infrastructure application marketplace; testing said distributed applications prior to

deployment; provisioning said distributed applications; and deprovisioning said distributed applications.

Vasell discloses this limitation. Per [19.5], Vasell discloses its **management components** providing an “**external interface**” [*application management portal*] [*application management portal*] **manage** the lifecycle operations of the “distributed” Java **service applications** implementing services in the **service gateway system**, including: (i) “**managing** the **creation** and **destruction** of the [application-]cells”; (ii) being “responsible for **starting or activating** [application-]cells... as well as and **stopping and destroying** [application-]cells...”; and (iii) “provid[ing] an **external interface** through which the **service applications**... may be **downloaded, installed, removed**, executed, and controlled” [bold emphasis indicating *provisioning said distributed applications; and deprovisioning said distributed applications*]. EX1004, 20:6-8, 20:32-34, 22:35-38, Annotated FIG. 6; EX1003, ¶¶526-529.

Claim 5

The system of claim 3, further comprising: a distributed resource service (DRS) which controls access to a plurality of resources upon receiving instructions from the application management portal.

Vasell discloses this limitation to POSITAs.

Per claim 14, POSITAs understood Vasell discloses that, as part of implementing Vasell’s “resource **management**”, enforcement of [application-]cell-

specific resource “quota[s]”, and “dynamic[] allocat[ion]” of “resource requirements” functions for the plurality of “distributed” **service applications** implementing services in the **service gateway system**, respective *instructions* would be provided *from* Vasell’s “**management system service**” having an “**external interface**” [*application management portal*] to respective **management software** (e.g., “**service boxlet[s]**”, “**diagnostic software**”) distributed between the **service gateway system servers** [*a distributed resource service (DRS) which is capable of ... configuring and managing platform resources*]. EX1003, ¶531.

As an example of Vasell’s **distributed management software** [*distributed resource service*], *upon receiving instructions* from Vasell’s “**management system service**” (e.g., upon receiving a network operator selection (and/or parameters establishing “a predetermined scheme of maximum allowed or minimum required resources, for a given situation” via the “**external interface** through which the **service applications** ... may be ... executed, and controlled”) [*from the application management portal*], *controlling access to a plurality of service gateway system resources*, Vasell discloses implementing “resource tradeoffs” for two or more “very resource-intensive” distributed **service applications** (e.g., a video telephone **service application** and a 3-D interactive **video game**) in the **service gateway system** “when one of the resource intensive **service applications**... is being used and someone seeks to start another.” EX1004, 12:51-54, 13:12-27, 12:27-29;

EX1016, 4, Figure 2, 7-9; *see also* [1.3] (describing Vasell’s disclosure of administrator-defined, [application-]cell-specific, “access rules” to respectively limit, monitor, and control available “resources (e.g., CPU, memory, persistent storage” to each “distributed” service application.) EX1003, ¶¶532-539.

Claim 7

The system of claim 5, wherein the DRS further includes a load controller adapted to monitor loads on at least one of the plurality of network device applications and at least one of the plurality of cloud applications and effect change in accordance with thresholds received from the application management portal.

Vasell alone, or in view of Rellermeier, discloses this limitation.

Per claims 5 and 14, Vasell discloses a “cell manager”, managed via the “management system service” having “an external interface”, controlling and providing instructions to respective, distributed management software (“service boxlet[s]”, “diagnostic software”), to (i) enforce [application-]cell-specific resource “quota[s]”, (ii) “handle[] resource management so that no single one of the [application-]cells... consumes too much of the service gateway system ... resources, e.g., CPU time, memory, persistent storage, and the like”, and (iii) “monitor”, “maintain and control” the distributed service applications implementing particular services in the service gateway system [*the DRS further includes a load controller adapted to monitor loads on at least one of the plurality*

of network device applications and at least one of the plurality of cloud applications]. EX1003, ¶541.

Vasell also discloses that the “**resource management**” and “**monitor[ing]**” functionality provided by this *distributed resource service* includes “resource requirements... be[ing] **dynamically allocated based upon a predetermined scheme of maximum allowed or minimum required resources**, for a given situation” [*the DRS further includes a load controller adapted to ... effect change in accordance with thresholds received from the application management portal*] so that “different ones of the **service applications**” do not “inadvertently interact in such a way that would cause them to... detrimentally affect each other”, and to “allow[] for resource tradeoffs between the **service applications**...”, which is the “controller executing... a... ‘load monitoring’ application” indicated in the ’823 Patent (§V.C) or equivalent thereof. *Id.*; EX1004, 12:51-63, 13:12-27; *see also* claim 15 (above), claim 8 (below) (identifying Vasell’s disclosure of *distributed content*, and *distributed notification, services*); EX1003, ¶¶542-550.

To the extent PO asserts this claim requires monitoring and control functionality for load (resource usage) expansion or contraction across multiple servers in Vasell’s **service gateway system** and that Vasell does not explicitly disclose this, POSITAs would have been motivated to modify Vasell’s teachings with Rellermeyer’s teachings of load balancing and elastic scalability, which

discloses the “controller executing... a ‘cloud breathing’ [**and**] ‘load monitoring’ application” indicated in the ’095 Patent (§V.C) or equivalent thereof, as discussed in the Motivation to Combine section. §VII.A.5 (EX1011, 2-5); EX1003, ¶¶551-555.

Claim 8

The system of claim 3 further comprising: a distributed notification service wherein notice of a predetermined event is sent from at least one of the plurality of network device applications to at least one of the plurality of cloud applications.

Vasell discloses this limitation. Specifically, Vasell discloses controlled implementation of “listeners” in the **service gateway system**, using “proxies” to send generated “event notifications... upon the occurrence... of a specified event”, and POSITAs understood that part of the secure communications between the distributed **service application** components (*see* [1.3]) would include these notifications of a specified event [*notice of a particular event*]. EX1004, 20:22-23, 19:21-38, 12:67-13:12; EX1003, ¶¶556-559.

Claim 18

The system of claim 1, wherein the at least one programmable network device and the programmable cloud device are adapted to verify the authenticity and integrity of updates to the plurality of network device applications and the plurality of cloud applications.

Vasell alone, or in view of Hall, discloses this limitation. Vasell discloses that authentication of upgrades to any **service application** is important due to the “open” nature (*see* [1.4], [19.6]) of the **service gateway system**. Specifically, Vasell discloses that, because “the **service gateway system** is accessed by and shared with multiple service providers and service gateway users”, “mechanisms for **authentication**... are integrated within the **service gateway system** design”, and “various **authorization and identity verification techniques**... [are] to be provided so that the implementation... associated with these services is robustly defended against intrusion and manipulation.” *Id.*; EX1004, 4:52-65, 11:59-12:10, 15:46-47; EX1016, 3-4, 6; EX1003, ¶¶563-566.

Vasell discloses “upgrade software” as a “**remote ... management**” operation, and describes an example of how the “**cell manager**” and “**cell table**” enable the implementation of services—“[e]ach service ... implemented as software **service applications**”—by the **service gateway system** to “continue undisturbed while [**service application**] configuration updates take place” [*updates to the plurality of network device applications and the plurality of cloud applications*]. EX1004, 5:16-26, 22:19-30, 22:6-18, FIG. 6, 12:11-29, 2:56-59; EX1016, 4; EX1003, ¶¶561-562.

Unsurprisingly, Vasell does not disclose the details of its **service gateway system servers** [*programmable network device and programmable cloud device*]

(see [1.1]-[1.2]) implementing well-known authentication, authorization, and/or identity verification security techniques for its intended “open” development environment of received distributed **service application software** by “different service providers”, “third party developers”, “nonspecialists” and “third party markets for... services”, especially where Vasell discloses that such distributed **service applications** are “Java **applications**” implemented in “a Java runtime in the form of a ... JVM” environment for which, as discussed in the State of the Art section, such security techniques were widely implemented by recipient devices. *Id.*; §V.F.2 (EX1034, 305; EX1044, 641-642, Figure 9-20, 643-644 (JDK 1.2); EX1050, 8-10); EX1003, ¶¶567, 133, 136, 140, 143-145.

Indeed, as described in the State of the Art section (§V.F.2), a standardized, well-known security scheme that was widely implemented by application developers and implementers for JVM-hosted applications, in this precise environment of developers with varying and unknown trustworthiness levels, was to require each developer to, using a unique private key issued to it by a trusted authority, “digitally sign” each of its respectively developed software applications (and corresponding software updates). *Id.* Then, prior to installation of each application/update, a **management agent** (and each “JVM security manager” of the JVMs in which the bundles are deployed on the OSs of the **servers**) would use the corresponding unique public key to verify that the respective developer is authentic

and the respective software/update is unmodified. *Id.*; §V.F.2 (EX1034, 304-306; EX1044, 641-644; EX1045, 169-172, 265-266; EX1057, 228, 432, 131, 145, 388-389, 122-123; EX1009, 457; EX1050, 8-10; EX1067); EX1003, ¶¶574, 133-147.

Thus, POSITAs understood that Vasell discloses this limitation based on its concerns regarding the authentication of any upgrades of its distributed Java **applications**, and its use of JVMs hosted on respective OSs of its **service gateway system servers**. *Id.*; EX1003, ¶¶576-577.

Moreover, to the extent PO asserts that Vasell does not explicitly disclose this limitation, it would have been obvious to modify Vasell's teachings with Hall's teachings of requiring OSGi-application-developers to "digitally sign" their respectively developed distributed **application components**, including updated versions, as discussed in the Motivation to Combine section, to achieve the benefits of rendering each **application** "tamperproof", determining the application-cell-specific-access-levels granted by Vasell's **management services** based on the respective developer's identity (signature), and to control the **application's** access to operations, information, services and **server** resources. §VII.A.5 (EX1009, 457-463, 330-334, 472-476, 179); EX1003, ¶¶578-579.

VIII. Discretionary Denial Would be Inappropriate

A. Discretionary Denial under the *Fintiv* Factors is Inappropriate

Under *Fintiv*, the Board takes a holistic view of whether efficiency, fairness, and the merits support the exercise of authority to deny institution in view of an earlier trial date in the parallel proceeding. *Apple Inc. v. Fintiv, Inc.*, Case IPR2020-00019, Paper 11, 6 (March 20, 2020).

According to the latest guidance from the Director:

The Board should first assess *Fintiv* factors 1–5; if that analysis supports discretionary denial, the Board should engage the compelling merits question. If the Board reaches the compelling merits analysis and finds compelling merits, it shall provide reasoning to explain its determination.

CommScope Techs. LLC, v. Dali Wireless Inc., IPR2022-01242, Paper 23, 6 (February 27, 2023).

Factor 1 is neutral because no request for stay has been filed in the Related Litigation.

Factor 2 favors institution. No trial date has been set in the Related Litigation, which was filed February 29, 2024 in W.D. Texas where, according to the most recent statistics on median time-to-trial for civil actions², the median time

² https://www.uscourts.gov/sites/default/files/2024-12/fcms_na_distprofile0930.2024.pdf.

from filing to trial is 33.1 months; accordingly, trial would be expected in late-November 2026. A Final Written Decision is expected in the present matter in September 2026, months before an expected trial in the Related Litigation.

Factor 3 is neutral. Fact discovery and claim construction are ongoing and, at the time of an expected institution decision, according to the operative Scheduling Order (EX1072), fact discovery will have closed and expert discovery will be ongoing.

Under factor 4, there will be no overlap of prior art issues as the Related litigation does not involve claims 3-5, 7-8, 12, 14-15 and 18 challenged herein, and Petitioner stipulates that, if trial is instituted, it will not pursue the same grounds in the litigation. *See Sand Revolution II, LLC v. Cont'l Intermodal Group-Trucking LLC*, IPR2019- 01393, Paper 24, 12 (PTAB June 16, 2020). Consequently, this factor favors institution.

Factor 5 is neutral because, although Petitioner is a defendant in the litigation, this is “far from an unusual circumstance” for petitioners in IPR proceedings. *Sand Revolution*, Paper 24, 12-13.

To the extent that the Board finds that factors 1-5 favor discretionary denial, under factor 6, other circumstances, including compelling merits, weigh in favor of institution. For example, Vasell is a parallel disclosure to the '823 Patent, meaning that it is directed at a substantially similar problem and proposes a substantially

similar solution as the '823 Patent, and discloses the same architecture and the same concepts disclosed in the '823 Patent, but uses some different words and leaves certain implementation details to design choices of POSITAs. Vasell's architecture and methods were foundational concepts (including implementation in a Java run-time environment) that were eventually standardized under OSGi. Technical implementation details that were intended to implement Vasell's teachings are described in each of the Ground 1 and Ground 2 references—Alves, Hall and Rellermeier. Accordingly, the motivation for POSITAs to modify the teachings of Vasell, to include the features described in these references for their intended purpose and to achieve predictable benefits with a reasonable expectation of success is demonstrated by the references themselves, thus demonstrating that the Petition is particularly strong in the underlying merits. The evidence presented, if unrebutted in trial, would plainly lead to a conclusion that one or more claims are unpatentable by a preponderance of the evidence.

B. Discretionary Denial Under §325(d) is Not Appropriate

Petitioner is unaware of any authority holding *Advanced Bionics* to be satisfied where (like here) the prosecution history lacks discussion of any reference relied upon in this petition. In such situations, the Board routinely does not decline institution on §325 grounds. Moreover, the challenges in this petition are non-cumulative because the prior art presented describes the claim limitations that were

added to gain allowance (including those expressly acknowledged by the patent to be well-known). See §V.A-§VII.A.6.

IX. Mandatory Notices Under 37 C.F.R. §42.8

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

The real party-in-interest in this Petition is Microsoft Corporation.

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

1. Judicial Matters

As of the filing date of this Petition and to the best knowledge of Petitioner, the '823 Patent is involved in the following litigations:

Edge Networking Systems, LLC v. Microsoft Corporation, Case No. 1:24-cv-215-DAE (W.D. Tex.) (filed Feb. 29, 2024) (ongoing); and

Edge Networking Systems LLC v. Amazon.Com, Inc., et al., Case No. EDTX-2-24-cv-00887 (E.D. Tex.) (filed Nov. 1, 2024) (ongoing).

Administrative Matters:

As of the filing date of this Petition and to the best knowledge of Petitioner, the '823 Patent has not been subject to any *inter partes* reviews, reissues, or reexaminations. Concurrently with the filing of this IPR petition, Petitioner is filing a petition for IPR of related U.S. Pat Nos. 10,686,871 (the "'871 Patent") and 10,893,095 (the "'095 Patent").

2. Related Patents

As of the filing date of this Petition and to the best knowledge of Petitioner, the following patent applications and patents are related to the '823 Patent:

| Patent/Publication Number | Application Number | Filing Date | Issue Date | Exhibit No. |
|---------------------------|--------------------|-------------|------------|-------------|
| 9,843,624 | 14/295,331 | 6/4/2014 | 12/12/2017 | EX1005 |
| 10,686,871 | 15/836,824 | 12/9/2017 | 6/16/2020 | EX1007 |
| 10,893,095 | 16/900,963 | 6/14/2020 | 1/12/2021 | EX1006 |
| 12,126,673 | 18/217,332 | 6/30/2023 | 10/22/2024 | |
| 12,113,850 | 18/600,742 | 3/10/2024 | 10/8/2024 | |
| 12,126,674 | 18/600,747 | 3/10/2024 | 10/8/2024 | |
| 12,113,851 | 18/664,279 | 5/14/2024 | 10/22/2024 | |
| | 18/824,854 | 9/4/2024 | | |

C. Lead/Back-up Counsel (37 C.F.R. § 42.8(b)(3)):

Lead Counsel:

Christopher J. Tyson, USPTO Reg. No. 63,850

DUANE MORRIS LLP, 901 New York Avenue N.W., Suite 700 East,
Washington, D.C. 20001

P: (202) 776-7851; F: (202) 478-2620; CJTyson@duanemorris.com

Back-Up Counsel:

John M. Baird, USPTO Reg. No. 57,585

DUANE MORRIS LLP, 901 New York Avenue N.W., Suite 700 East,
Washington, D.C. 20001

P: (202) 776-7819; F: (202) 478-2620; JMBaird@duanemorris.com

Glenn D. Richeson, USPTO Reg. No. 73,480

DUANE MORRIS LLP, 1075 Peachtree Street NE, Suite 1700
Atlanta, GA 30309

P: (404) 253-6993; F: (404) 759-2703; GDRicheson@duanemorris.com

Patrick D. McPherson, USPTO Reg. No. 46,255

DUANE MORRIS LLP, 901 New York Avenue N.W., Suite 700 East,
Washington, D.C. 20001

P: (202) 776-5214; F: (202) 478-0826; PDMcPherson@duanemorris.com

D. Notice of Service Information (37 C.F.R. § 42.8(b)(4)):

Please direct all correspondence to lead and back-up counsel at the above addresses. Petitioner consents to electronic service at the email addresses above.

X. CONCLUSION

Petitioner requests the Board institute IPR and cancel the Challenged Claims.

Respectfully submitted,

DUANE MORRIS LLP

/Christopher J. Tyson/

Christopher J. Tyson, Reg. No. 63,850

901 New York Avenue, N.W.

Suite 700 East

Washington, D.C. 20001-4795

P: (202) 776-7851

F: (202) 478-2620

CJTyson@duanemorris.com

ATTORNEY FOR PETITIONER

Dated: February 17, 2025

Petition for *Inter Partes* Review of U. S. Patent No. 11,695,823

P: (202) 776-7851

F: (202) 478-2620

CJTyson@duanemorris.com

ATTORNEY FOR PETITIONER