

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMAZON.COM, INC., AMAZON WEB SERVICES, INC.,
and AMAZON.COM SERVICES LLC,

Petitioner

v.

B.S.D. CROWN, LTD.,

Patent Owner.

IPR2025-00057

U.S. Patent No. 8,934,887

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 8,934,887**

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

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	GROUND FOR STANDING.....	1
III.	IDENTIFICATION OF CHALLENGE	1
	A. Prior Art.....	1
	B. Grounds for Challenge (all §103).....	2
IV.	'887 PATENT OVERVIEW	2
	A. Specification.....	2
	B. Prosecution History	3
V.	LEVEL OF ORDINARY SKILL	5
VI.	CLAIM CONSTRUCTION	5
VII.	DETAILED EXPLANATION OF GROUNDS.....	5
	A. Ground 1A: Yoon (Claims 1-3, 5, 20-21, 28).....	5
	1. Yoon.....	5
	2. Claim 1	8
	3. Claim 2	12
	4. Claim 3	13
	5. Claim 5	14
	6. Claim 20	15
	7. Claim 21	16
	8. Claim 28	16
	B. Ground 1B: Yoon, Nix (claims 4, 14, 22).....	19
	1. Nix.....	19
	2. Motivation to Combine	20
	3. Claim 4	22
	4. Claim 14	23
	5. Claim 22	23

TABLE OF CONTENTS
(continued)

	Page
C. Ground 1C: Yoon, Lubonski, Richardson (claims 6-8, 11-13, 15-17, 23-25).....	23
1. Lubonski.....	23
2. Richardson	26
3. Motivation to Combine	28
4. Claim 6.....	30
5. Claim 7.....	33
6. Claim 8.....	33
7. Claim 11	34
8. Claim 12.....	35
9. Claims 13, 15-17, 23-25.....	35
D. Ground 2A: Overton (claims 1-3, 5, 20-21, 28).....	36
1. Overton.....	36
2. Claim 1	38
3. Claim 2.....	44
4. Claim 3.....	45
5. Claim 5.....	45
6. Claim 20.....	46
7. Claim 21	47
8. Claim 28.....	48
E. Ground 2B: Overton, Nix (claims 4, 14, 22).....	52
1. Motivation to Combine	52
2. Claim 4.....	56
3. Claim 14.....	57
4. Claim 22.....	58
F. Ground 2C: Overton, Lubonski, Richardson (claims 6-8, 11-13, 15-17, 23-25).....	58

TABLE OF CONTENTS
(continued)

	Page
1. Motivation to Combine	58
2. Claim 6	61
3. Claim 7	62
4. Claim 8	62
5. Claim 11	63
6. Claim 12	64
7. Claims 13, 15-17, 23-25	64
G. Ground 3A: Lu (claims 1-3, 5, 20-21)	65
1. Lu	65
2. Claim 1	67
3. Claim 2	73
4. Claim 3	73
5. Claim 5	74
6. Claim 20	74
7. Claim 21	75
H. Ground 3B: Lu, Lubonski, Richardson (claims 6-8, 11-13, 15-17, 23-25)	76
1. Motivation to Combine	76
2. Claim 6	78
3. Claim 7	79
4. Claim 8	80
5. Claim 11	81
6. Claim 12	81
7. Claims 13, 15-17, 23-25	82
VIII. DISCRETIONARY DENIAL FACTORS FAVOR INSTITUTION	82
IX. CONCLUSION	83

TABLE OF CONTENTS
(continued)

	Page
X. MANDATORY NOTICES	83
A. Real Party in Interest	83
B. Related Matters.....	84
C. Notice of Counsel and Service Information.....	84

EXHIBIT LIST

Exhibit	Reference
1001	U.S. Patent No. 8,934,887 (“’887 Patent” or “’887Pat.”)
1002	File History of the ’887 Patent (“’887FH”)
1003	Declaration of Dr. Paul Martin (“Martin”)
1004	Dr. Paul Martin CV
1005	Korean Patent Application Publication No. 10-2012-0046807 (“Yoon”)
1006	U.S. Patent No. 9,424,052 (“Overton”)
1007	U.S. Patent Application Publication No. 2009/0305790 A1 (“Lu”)
1008	M. Lubonski, V. Gay and A. Simmonds, “An adaptation architecture to improve QoS of multimedia services for enterprise remote desktop protocols,” Next Generation Internet Networks, 2005, Rome, Italy, 2005, pp. 149-156 (“Lubonski”)
1009	Richardson, Iain E., The H.264 Advanced Video Compression Standard (2nd ed.), 2010, Wiley Publishing (“Richardson”)
1010	U.S. Patent Application Publication No. 2006/0098619 A1 (“Nix”)
1011	Declaration of June Ann Munford
1012	Complaint, <i>BSD Crown, Ltd. f/k/a Emblaze, Ltd. v. Amazon.com Inc.</i> , NDCA-3-24-cv-03503
1013	Plaintiff’s Infringement Contentions in <i>BSD Crown, Ltd. f/k/a Emblaze, Ltd. v. Amazon.com Inc.</i> , NDCA-3-24-cv-03503
1014	Excerpts from Federal Court Management Statistics, June 2024
1015	U.S. Patent Application Publication No. 2001/0052012 A1 (“Rinne”)
1016	M. A. Talaat, M. A. Koutb, H. M. Kelash and R. H. Aboelez, “Content-aware adaptive video streaming system,” 2005 International Conference on Information and Communication Technology, Cairo, 2005, pp. 265-276 (“Talaat”)

Exhibit	Reference
1017	A Few More Reasons Why RDP is Insecure. https://thehackernews.com/2023/07/a-few-more-reasons-why-rdp-is-insecure.html
1018	Understanding the Remote Desktop Protocol (RDP). https://learn.microsoft.com/en-us/troubleshoot/windows-server/remote/understanding-remote-desktop-protocol
1019	Remote Desktop Services overview in Windows Server. https://learn.microsoft.com/en-us/windows-server/remote/remote-desktop-services/remote-desktop-services-overview
1020	Remote Desktop clients for Remote Desktop Services and remote PCs. https://learn.microsoft.com/en-us/windows-server/remote/remote-desktop-services/clients/remote-desktop-clients
1021	Remote takeover: How RealVNC conquered the world. https://www.cam.ac.uk/research/features/remote-takeover-how-realvnc-conquered-the-world
1022	Download VNC Server by RealVNC. https://www.realvnc.com/en/connect/download/vnc/
1023	Download VNC Viewer by RealVNC. https://www.realvnc.com/en/connect/download/viewer/
1024	Robert. P. Goldberg. Survey of virtual machine research. IEEE Computer Magazine, 7(6):34-45, 1974. https://web.eecs.umich.edu/~prabal/teaching/eecs582-w11/readings/Gol74.pdf
1025	R. J. Creasy. The Origin of the VM/370 Time-Sharing System. IBM Journal of Research and Development, 25(5):483-490, September 1981. https://pages.cs.wisc.edu/~stjones/proj/vm_reading/ibmrd2505M.pdf
1026	VMware Celebrates Seven Years of Continual Innovation and Execution. https://news.broadcom.com/releases/7year

Exhibit	Reference
1027	[Xen-devel] [ANNOUNCE] Xen high-performance x86 virtualization. https://sourceforge.net/p/xen/mailman/message/5533663/
1028	News (older entries). https://www.virtualbox.org/wiki/News_pre_4_3
1029	Ten years of KVM. https://lwn.net/Articles/705160/
1030	Hyper-V Made Available. https://adtmag.com/articles/2008/06/26/hyperv-made-available.aspx
1031	Amazon EC2 beta. https://aws.amazon.com/blogs/aws/amazon_ec2_beta/
1032	Microsoft launches Windows Azure. https://news.microsoft.com/announcement/microsoft-launches-windows-azure/
1033	Reflecting on our ten year App Engine journey. https://cloud.google.com/blog/products/gcp/reflecting-on-our-ten-year-app-engine-journey/
1034	Guide to Security for Full Virtualization Technologies. https://doi.org/10.6028/NIST.SP.800-125
1035	Review: OnLive -- The What, Why and Who of Gaming in the Cloud. https://www.wired.com/2011/06/review-onlive-the-what-why-and-who-of-gaming-in-the-cloud/
1036	Gaikai is Live! https://dperry.com/2011/02/06/gaikai_is_live/
1037	Qi, H., & Gani, A. (2012, May). Research on mobile cloud computing: Review, trend and perspectives. In 2012 second international conference on digital information and communication technology and it's applications (DICTAP) (pp. 195-202). iee. https://arxiv.org/pdf/1206.1118
1038	Kumar, Karthik, and Yung-Hsiang Lu. Cloud computing for mobile users: Can offloading computation save energy?. Computer 43, no. 4 (2010): 51-56. https://www.cs.purdue.edu/homes/bb/cs590/handouts/computer.pdf

Exhibit	Reference
1039	Dinh, H.T., Lee, C., Niyato, D. and Wang, P. (2013), A survey of mobile cloud computing: architecture, applications, and approaches. Wirel. Commun. Mob. Comput., 13: 1587-1611. https://doi.org/10.1002/wcm.1203

LISTING OF CHALLENGED CLAIMS

Reference	Claim Limitation
Claim 1	
1[pre]	A method for emulating a mobile device at a server, the method comprising:
1[a]	receiving, by the server, input information from at least one mobile device, wherein the input information has not been processed to produce a display, audio and a hardware (HW) action by a HW component of the mobile device;
1[b]	processing, by a mobile device operating system (OS) executed at the server, the input information to produce a stream of user interface information for use in reproducing at the mobile device at least one of the display, the audio and the hardware (HW) action by the HW component of the mobile device based on a same instance of the mobile device operating system; and
1[c]	sending, by the server, the stream of user interface information to the mobile device to reproduce at the mobile device at least one of the display, the audio and the HW action by the HW component of the mobile device.
Claim 2	
2	The method of claim 1, further comprising: emulating, by the server, mobile device HW and executing the mobile device OS on the emulated mobile device HW.
Claim 3	
3[a]	The method of claim 1, further comprising: executing, by the server, at least one application on the mobile OS;
3[b]	wherein the at least one application processes the input information to produce the stream of user interface information.

Reference	Claim Limitation
Claim 4	
4	The method of claim 1, further comprising: providing, by the server, a mobile device gateway for handling and providing internet access, voice calls and a gateway to a public switched telephone network (PSTN).
Claim 5	
5	The method of claim 1, further comprising: receiving, by the server, the input information from a plurality of different mobile devices, wherein a same instance of the mobile device OS processes the input information from each of the plurality of different mobile devices to produce the stream of user interface information.
Claim 6	
6	The method of claim 1, wherein the stream of user interface information comprises a plurality of different types of data and the server sends each different type of data to the mobile device in a separate data stream.
Claim 7	
7	The method of claim 6, wherein at least one of said separate data streams comprises a quality of service different from a quality of service of another of said separate data streams.
Claim 8	
8	The method of claim 6, wherein at least one of said separate data streams comprises an encoding method and a bandwidth different from an encoding method and a bandwidth of another of said separate data streams.

Reference	Claim Limitation
Claim 11	
11[pre]	A method for emulating a mobile device at a server, the method comprising:
11[a]	receiving, by the server, input information from at least one mobile device;
11[b]	processing, by a mobile device operating system (OS) executed at the server, the input information to produce a stream of user interface information for use in reproducing at the mobile device at least one of a display, audio and a hardware (HW) action by a HW component of the mobile device based on a same instance of the mobile device OS; and
11[c]	sending, by the server, the stream of user interface information to the mobile device, wherein the stream of user interface information from the same instance of the mobile device OS comprises a plurality of different types of data and the server sends each different type of data to the mobile device in a separate data stream for reproduction on the mobile device of at least one of the display, audio and the HW action by the HW component of the mobile device.
Claim 12	
12	The method of claim 11, further comprising: emulating, by the server, mobile device HW and executing the mobile device OS on the emulated mobile device HW.
Claim 13	
13[a]	The method of claim 11, further comprising: executing, by the server, at least one application on the mobile device OS;
13[b]	wherein the at least one application processes the input information to produce the stream of user interface information.

Reference	Claim Limitation
Claim 14	
14	The method of claim 11, further comprising: providing, by the server, a mobile device gateway for handling and providing internet access, voice calls and a gateway to a public switched telephone network (PSTN).
Claim 15	
15	The method of claim 11, further comprising: receiving the input information from a plurality of different mobile devices, wherein a same instance of the mobile device OS processes the input information from the plurality of different mobile devices to produce the stream of user interface information.
Claim 16	
16	The method of claim 11, wherein at least one of said separate data streams comprises a quality of service different from a quality of service of another of said separate data streams.
Claim 17	
17	The method of claim 11, wherein at least one of said separate data streams comprises an encoding method and a bandwidth different from an encoding method and a bandwidth of another of said separate data streams.
Claim 20	
20[pre]	A method for emulating a mobile device at a server, the method comprising:
20[a]	receiving, by the server, input information from a plurality of different mobile devices, wherein the input information has not been processed to produce a display, audio and a hardware (HW) action by a corresponding HW component of the mobile devices;

Reference	Claim Limitation
20[b]	processing, by a same instance of a mobile device operating system (OS) executed at the server, the input information from each of the plurality of different mobile devices to produce user interface information for use in reproducing at the different mobile devices at least one of the display, audio and the HW action by the HW components of the different mobile device based on the same instance of the mobile device OS; and
20[c]	sending, by the server, the user interface information to at least one of the plurality of different mobile devices.
Claim 21	
21[a]	The method of claim 20, further comprising: executing, by the server, at least one application on the mobile device OS;
21[b]	wherein the at least one application processes the input information to produce the user interface information.
Claim 22	
22	The method of claim 20, further comprising: providing, by the server, a mobile device gateway for handling and providing internet access, voice calls and a gateway to a public switched telephone network (PSTN).
Claim 23	
23	The method of claim 20, wherein the stream of user interface information comprises a plurality of different types of data and the server sends each different type of data to the mobile device in a separate data stream.

Reference	Claim Limitation
Claim 24	
24	The method of claim 23, wherein at least one of said separate data streams comprises a quality of service different a quality of service of another of said separate data streams.
Claim 25	
25	The method of claim 23, wherein at least one of said separate data streams comprises an encoding method and a bandwidth different from an encoding method and a bandwidth of another of said separate data streams.
Claim 28	
28[pre]	A method for emulating a mobile device at a server, the method comprising:
28[a]	receiving, by server, first input information from a mobile device;
28[b]	processing, by a first mobile device operating system (OS) executed at the server, the first input information to produce a first stream of user interface information for use in reproducing at the mobile device at least one of a display, audio and a hardware (HW) action by a HW component of the mobile device; and
28[c]	sending, by the server, the first stream of user interface information to the mobile device to reproduce on the mobile device at least one of the display, audio and the HW action by the HW component of the mobile device;
28[d]	receiving, by the server, second input information from the mobile device;
28[e]	processing, by a second mobile device operating system (OS) executed by the server, the second input information to produce a second stream of user interface information for reproducing at least

Reference	Claim Limitation
	one of a display, audio and a HW action by a HW component of the mobile device, wherein the second mobile device OS has at least one of a different OS type, a different OS version and a different OS instance than the first mobile device OS; and
28[f]	sending, by the server, the second stream of user interface information to the mobile device to reproduce on the mobile device at least one of the display, audio and the HW action by the HW component of the mobile device.

I. INTRODUCTION

Amazon.com, Inc., Amazon Web Services, Inc., and Amazon.com Services LLC (“Amazon” or “Petitioner”) request *inter partes* review (“IPR”) of claims 1-8, 11-17, 20-25, and 28 (“Challenged Claims”) of U.S. Patent No. 8,934,887 (“the ’887 Patent” or “’887pat.”; EX-1001).

II. GROUNDS FOR STANDING

Petitioner certifies the ’887 Patent is available for IPR, and Petitioner is not barred or estopped from requesting IPR on the grounds herein. Petitioner files this Petition within one year of service on Amazon of PO’s district court complaint (EX-1012).

III. IDENTIFICATION OF CHALLENGE

A. Prior Art

Petitioner respectfully requests cancellation of the Challenged Claims based on:

Prior Art
KR 2012/0046807 (“Yoon”); published May 11, 2012; prior art under 35 U.S.C. §102(a); EX1005
U.S. Patent 9,424,052 (“Overton”); filed May 24, 2011; prior art under 35 U.S.C. §102(e); EX1006
U.S. Published Patent Application 2009/0305790 (“Lu”); published December 10, 2009; prior art under 35 U.S.C. §102(b); EX1007
U.S. Published Patent Application 2006/0098619 (“Nix”); published May 11, 2006; prior art under 35 U.S.C. §102(b); EX1010

M. Lubonski, V. Gay and A. Simmonds, “An adaptation architecture to improve user-perceived QoS of multimedia services for enterprise remote desktop protocols,” Next Generation Internet Networks, 2005, Rome, Italy, 2005, pp. 149-156 (“Lubonski”); Presented April 18-20, 2005; prior art under 35 U.S.C. §102(b)¹; EX1008

Richardson, Iain E., The H.264 Advanced Video Compression Standard (2nd ed.), 2010, Wiley Publishing (“Richardson”); prior art under 35 U.S.C. §102(b)²; EX1009

B. Grounds for Challenge (all §103)

Ground	Claims	Reference(s)
1A	1-3, 5, 20-21, 28	Yoon
1B	4, 14, 22	Yoon, Nix
1C	6-8, 11-13, 15-17, 23-25	Yoon, Lubonski, Richardson
2A	1-3, 5, 20-21, 28	Overton
2B	4, 14, 22	Overton, Nix
2C	6-8, 11-13, 15-17, 23-25	Overton, Lubonski, Richardson
3A	1-3, 5, 20-21	Lu
3B	6-8, 11-13, 15-17, 23-25	Lu, Lubonski, Richardson

IV. '887 PATENT OVERVIEW

A. Specification

The '887 Patent discloses “decoupling mobile device HW [hardware] capabilities from actual mobile device performance and a user experience, which enables upgrading mobile device functionality, performance, OS and applications without the need to change the HW or the local SW [software] of the mobile device

¹ Publicly available by May 27, 2005. EX1011.

² Publicly available by November 15, 2010. EX1011.

itself.” ’887pat., 2:20-24. The mobile device “receives input information from the input components” on the local HW “and sends the input information to the server 12 for processing by the server” (*id.*, 4:55-57) after encoding for transmission (*id.*, 7:19-25). The mobile device thus does not process the input information to produce a display, audio, or HW action by the mobile device. *Id.*, 4:57-60. “[T]he server 12 executes a mobile device operating system...and one or more applications...on emulated mobile device HW based on the input information received from the mobile device [and] streams all animations and/or graphics changes...that occur on the virtual mobile device...to the mobile device.” *Id.*, 4:66-5:8. “The user interface information is data that can be used to recreate or produce a display and/or audio without any processing thereof other than the decoding and playback or reproduction of the user interface information itself by the mobile device.” *Id.*, 5:8-12.

B. Prosecution History

The ’887 Patent issued from U.S. Application 13/606,675 (“’675 Application”). Relevant here, after an obviousness rejection, the Applicant amended independent claims 20, 30, and 47 (issued as claims 1, 11, and 20, respectively) to recite that the server produces “a stream of user interface information” for use in reproducing a display, audio, or HW action “at the mobile device.” 887FH, 182-190 (emphasis original). The last independent claim (39, which issued as claim 20) was

amended to specify that the server produces user interface information “based on the same instance of the mobile device OS.” *Id.*, 187.

Regarding claim 20 (issued claim 1), Applicant argued that “all input data processing functionality is executed at the server, and the local, mobile physical device itself does not execute any such functionality.” *Id.*, 195 (emphasis original).

Regarding claim 30 (issued claim 11) and 39 (issued claim 20), Applicant argued that the “claimed invention provides multiple data streams (e.g., audio, video, compressed text, voice and/or sensors) arriving from a single mobile device emulation.” *Id.*, 198.

Applicant also argued that “the same instance of a virtual mobile device serves multiple mobile devices.” *Id.*, 198-199 (emphasis original).

Regarding claim 47 (issued claim 28), Applicant argued that “different kinds of mobile device OS’s are executed and not merely multiple instances of the same mobile device OS.” *Id.*, 199 (emphasis original).

The claims were then allowed following an Examiner’s Amendment to claims 20 and 39. *Id.*, 211-214. Claim 20 (issued claim 1) was amended to recite “at least one of the display, the audio, and the hardware (HW) action by the HW component of the mobile device based on a same instance of the mobile device operating system,” and claim 39 (issued claim 20) was amended to recite “wherein the input information has not been processed to produce a display, audio and a hardware (HW)

action by a corresponding HW component of the mobile devices.” *Id.* (emphasis original). As discussed below, each of the limitations added during prosecution was known in the prior art.

V. LEVEL OF ORDINARY SKILL

A person of ordinary skill in the art (“POSITA”) at the time of the ’887 Patent’s alleged invention would have had a bachelor’s degree and two years’ experience in cloud computing or a similar field, or a master’s degree, in computer science or similar field. Martin, ¶61.

VI. CLAIM CONSTRUCTION

Claims should be construed “in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.” 37 C.F.R. §42.100(b); *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). No explicit construction is necessary to determine that the Challenged Claims are invalid, and the petition applies the plain and ordinary meanings of the claim terms.

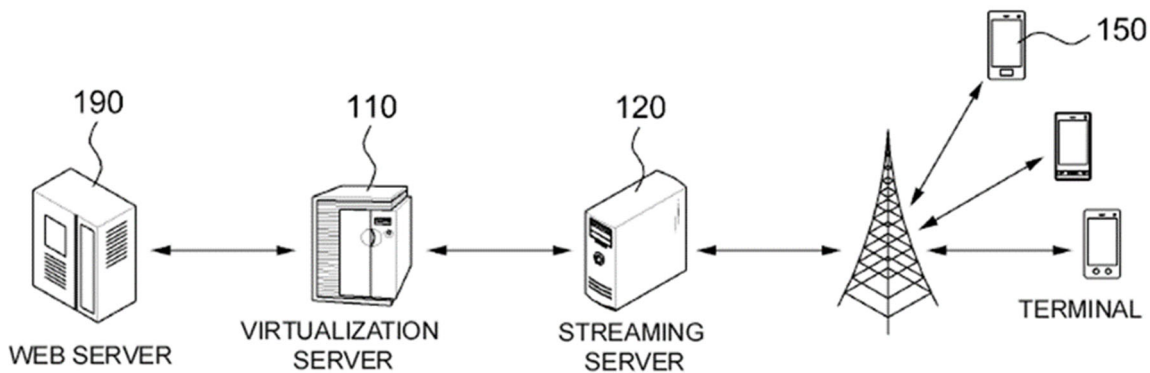
VII. DETAILED EXPLANATION OF GROUNDS

A. Ground 1A: Yoon (Claims 1-3, 5, 20-21, 28)

1. Yoon

Yoon discloses “a method and an apparatus for providing a virtual environment of a mobile operating system (OS) and a mobile application.” Yoon,

Abstract. Yoon teaches a system comprising terminal 150 (such as a smart phone), virtualization server 110, streaming server 120, and web server 190, all “connected through a high-speed network such as a local area network.” Yoon, 11.

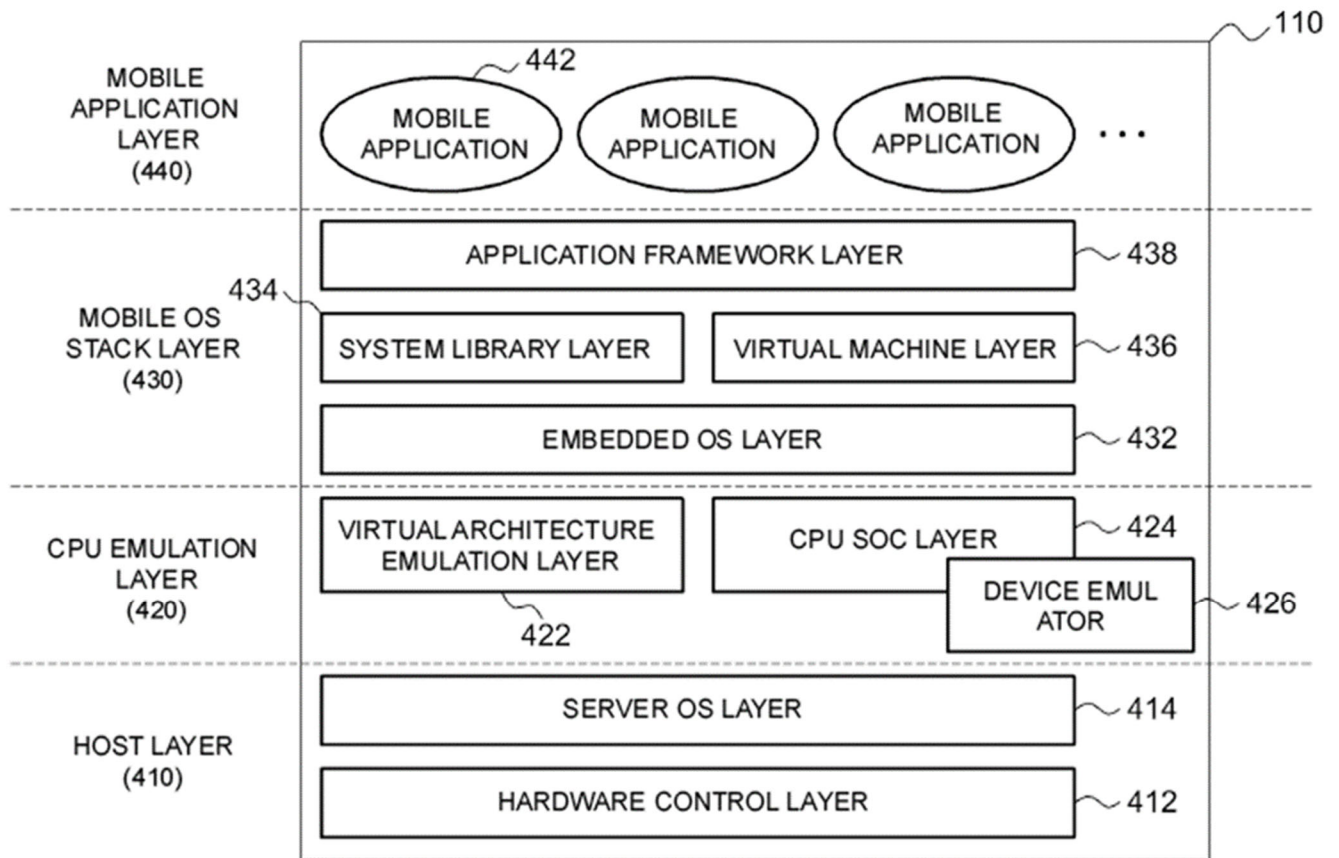


Yoon, Fig. 1.

Terminal 150 comprises thin client 150 that “interprets data related to execution of the mobile OS and/or application transmitted by the virtualization server 110...and outputs the data to the display or speaker, etc., of the terminal.” Yoon, 12. The thin client also transmits input information from the terminal 150 “to the virtualization server 110 to allow the mobile OS and/or application [executed at virtualization server 110] to operate according to the input information.” *Id.*

Yoon’s virtualization server “provides a virtual environment of a mobile OS and an application operating in the mobile OS.” Yoon, 7. The virtualization server further includes “a CPU emulation layer configured to emulate a CPU for supporting the mobile OS” and “a device emulator configured to emulate a device of a terminal

having the CPU mounted therein.” *Id.*, 10. “The streaming server 120 performs a relay role of data transmission and reception between the virtualization server 110 and the terminal 150.” *Id.*, 11. “[T]he functions of the streaming server 120...may be integrated into the functions of the virtualization server 110.” *Id.*, 15. “The web server 190 provides the web service by...operating in a virtual environment of the virtualization server 110.” *Id.*, 11. “[T]he functions of the web server 190...may be integrated into the functions of the virtualization server 110” as well. *Id.*, 16-17. In this Petition, Petitioners map to an embodiment of Yoon where the functions of the web server and streaming server are integrated into the virtualization server. Martin, ¶75. Figure 4 depicts Yoon’s virtualization server in additional detail:



Yoon, Fig. 4.

2. Claim 1

a. 1[pre]

Yoon discloses or suggests 1[pre]. Martin, ¶79.

Yoon discloses “a method for the virtualization server 110 to deliver information on the operating mobile OS and/or application to the terminal 150.” Yoon, 13. Yoon further discloses that its “device emulator 426...emulate[s] a device of a terminal having a mobile terminal CPU mounted therein.” Yoon, 21. Yoon thus

discloses a method for its virtualization server (“server”) “*emulating*” a terminal (“*mobile device*”). Martin, ¶79.

b. 1[a]

Yoon discloses or suggests 1[a]. Martin, ¶¶80-82.

Yoon’s virtualization server “receiv[es] input information for manipulating the mobile OS or the application from the terminal.” Yoon, 8, *see also, id.*, 12 (“The user of the terminal 150 may manipulate the mobile OS and/or application...by using...an input device...of the terminal 150”), 14-16 (explaining the virtualization server transforms, i.e., processes, input information from the terminal). Yoon does not mention processing these inputs at the terminal, and given that the input information is expressly “for manipulating the mobile OS or the application” at the virtualization server, a POSITA would have understood Yoon’s input information is processed at the server rather than being processed at the terminal (“*the input information has not been processed to produce a display, audio and a hardware (HW) action by a HW component of the mobile device*”). Martin, ¶¶81-82; *see also*, Yoon, 14-16 (processing performed at virtualization server) 18-19, Fig. 2, 20 (“controller 320 may process the input information. The processed input information may be reflected on operations of the mobile OS or the application.”).

c. 1[b]

Yoon discloses or suggests 1[b]. Martin, ¶¶83-85.

Yoon discloses that its virtualization server executes a mobile device operating system. Yoon, 11 (“mobile OS and an application operating in the mobile OS are executed [at the virtualization server] as if they are directly executed by the terminal 150”), 11-12 (“The virtualization server 110 operates the mobile OS and the application and provides information on the operating mobile OS and/or application to the terminal 150, thereby providing a user of the terminal 150 with the environment in which the mobile OS and/or application is executed as if it is directly executed by the terminal 150.”), 18 (“the mobile OS and/or the application executed in the virtualization server 110...”), 22 (“the device emulator may emulate an execution screen of the mobile OS and/or the application”); Martin, ¶83.

Yoon further discloses the user “manipulate[s] the mobile OS and/or application provided by the virtualization service by using an output device (for example, a liquid crystal display (LCD), a speaker, etc.) and an input device...of the terminal 150,” and the “virtualization server 110 operates the mobile OS and the application and provides information on the operating mobile OS and/or application to the terminal 150.” Yoon, 11-12. For example, “the virtualization server 110 generates an execution screen image of the running mobile OS and/or application” as well as “a sound generated by the running mobile OS and/or application,” and both are transmitted in a stream to the terminal. Yoon, 14-15. Yoon thus discloses

the mobile device OS executed at the virtualization server processes input from terminal 150 to produce a stream of user interface information. Martin, ¶84.

Finally, a POSITA would have understood that the stream of user input information from the virtualization server is “*based on a same instance of the mobile device operating system*” because, while Yoon contemplates either a single emulation instance servicing multiple terminals or a single emulation instance servicing a single terminal, Yoon never discusses multiple emulation instances providing user interface information to the same terminal. Martin, ¶85; *see also*, Yoon, 22 (explaining a single CPU emulation layer “may...provide a virtual environment to one or more virtualization service requestors (for example, the user of the terminal 150)” and “a separate CPU emulation layer 420 and/or mobile OS stack layer 430 may be generated for each virtualization service requestor”). Nor would it make sense for multiple emulations to provide user interface information to the same terminal because it would unnecessarily increase overhead by requiring duplication of transmissions and processing of user inputs. Martin, ¶85; *see* ’887FH, 198.

d. 1[c]

Yoon discloses or suggests 1[c]. Martin, ¶86.

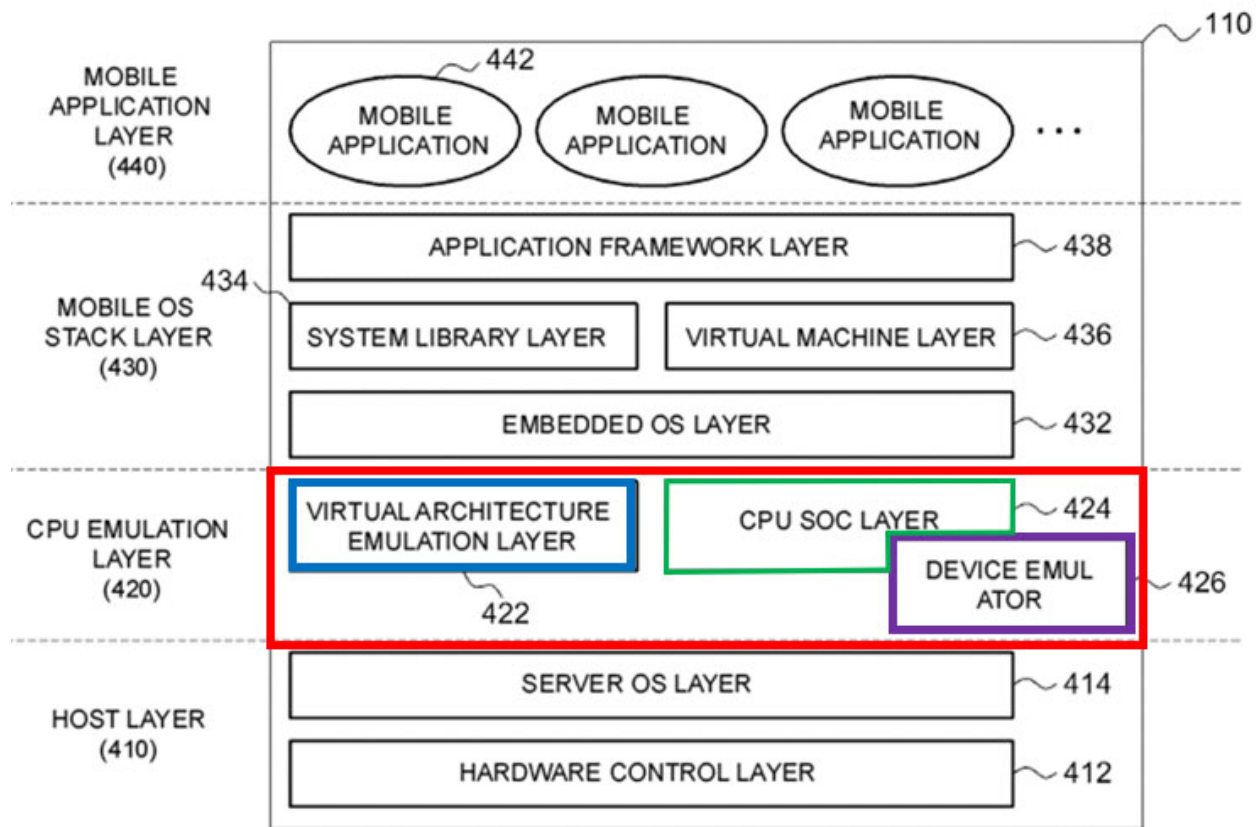
Yoon discloses that the virtualization server transmits (“*sending*”) the stream of video and sound (“*user interface information*”) output from the virtualization

server to the terminal (“*the mobile device*”). Yoon, 14-15. The terminal then displays images and plays sounds received from the virtualization server (“*reproduce at the mobile device at least one of the display, the audio, and the HW action...*”). Yoon, 18 (“the terminal 150 outputs the virtual environment to the output device by using the received virtual environment information”). Martin, ¶86.

3. Claim 2

Yoon discloses or suggests this claim. Martin, ¶87.

Yoon discloses that its “CPU emulation layer 420 [red] emulates a CPU for supporting a mobile OS.” Yoon, 21. Yoon further discloses a “virtual architecture emulation layer 422” [blue], a “virtual CPU system on chip (SOC) layer 424” [green], and a “device emulator 426” [purple] “plugged in[to] the virtual CPU SOC layer 424 to emulate a device of a terminal having a mobile terminal CPU mounted therein” (collectively “*emulating, by the server, mobile device HW*”). Yoon, 21, Fig. 4. The mobile device OS is then executed on the emulated mobile device HW. *Id.* (“mobile OS stack layer 430 may operate as an application of the...CPU emulation layer 420, and may provide the function of the mobile OS”); Martin, ¶87.



Yoon, Figure 4³.

4. Claim 3

a. 3[a]

Yoon discloses or suggests 3[a]. Martin, ¶88.

Yoon discloses that “an application 422 of the mobile OS is executed in the mobile application layer 440” of the virtualization server. Yoon, 21. “The mobile application layer 440 operates as an application of the mobile OS stack layer 430,”

³ All annotations to figures added, unless otherwise noted.

which “provide[s] the function of the mobile OS.” *Id.* Thus, the server executes “*at least one application on the mobile OS.*” Martin, ¶88.

b. 3[b]

Yoon discloses or suggests this limitation. Martin, ¶89.

As explained in Section VII.A.2.c, Yoon teaches that, at the server, the mobile OS *and/or*⁴ application processes user input to produce a stream of interface information. Thus, Yoon discloses the “*at least one application processes user input to produce the stream of user interface information.*” Martin, ¶89.

5. Claim 5

Yoon discloses or suggest this claim. Martin, ¶90.

Yoon discloses “[e]ach of the plurality of mobile OS stack layers 430 may be to provide a virtual environment for one or more virtualization service requestors.” Yoon, 22. Thus “each” mobile OS stack layer can provide a virtual environment to more than one virtualization service requestor (e.g., multiple terminals 150), and a POSITA would therefore have understood Yoon teaches the “*same instance of the mobile OS*” on the virtualization server can receive and “*process the input information*” from a “*plurality of different mobile devices to produce the stream of*

⁴ All emphasis added, unless noted otherwise.

user interface information” for each (as discussed in claim 1, Section VII.A.2).
Martin, ¶90.

6. Claim 20

a. 20[pre]

Yoon discloses or suggests 20[pre] for the reasons discussed in Section VII.A.2.a. Martin, ¶91.

b. 20[a]

Yoon discloses or suggests 20[a] for the reasons discussed in Sections VII.A.2.b, VII.A.5. Martin, ¶92. Yoon’ virtualization server receives input information that has not been processed from a terminal (Section VII.A.2.b), and multiple terminals may interact with the same mobile OS instance (Section VII.A.5).

c. 20[b]

Yoon discloses or suggests 20[b] for the reasons discussed in Sections VII.A.2.c, VII.A.5. Martin, ¶93. Yoon teaches that the same mobile OS instance processes input information from multiple terminals, §VII.A.5, and produces a media stream to send to each of the terminals, where the stream will be rendered, §VII.A.2.c.

d. 20[c]

Yoon discloses or suggests 20[c] for the reasons discussed in Sections VII.A.2.d, VII.A.5. Yoon discloses the same instance of the mobile OS can provide

a virtual environment to multiple terminals, so a POSITA would have understood the mobile OS instance sends user interface information to at least one of said terminals (“*at least one of the plurality of different mobile devices*”). Martin, ¶94.

7. Claim 21

a. 21[a]

Yoon discloses or suggests 21[a] for the reasons discussed in Section VII.A.4.a. Martin, ¶95.

b. 21[b]

Yoon discloses or suggests 21[b] for the reasons discussed in Section VII.A.4.b. Martin, ¶96

8. Claim 28

a. 28[pre]

Yoon discloses or suggests 28[pre] for the reasons discussed in Section VII.A.2.a. Martin, ¶97.

b. 28[a]

Yoon discloses or suggests 28[a] for the reasons discussed in Section VII.A.2.b. Martin, ¶98.

c. 28[b]

Yoon discloses or suggests 28[b] for the reasons discussed in Section VII.A.2.c. Martin, ¶99.

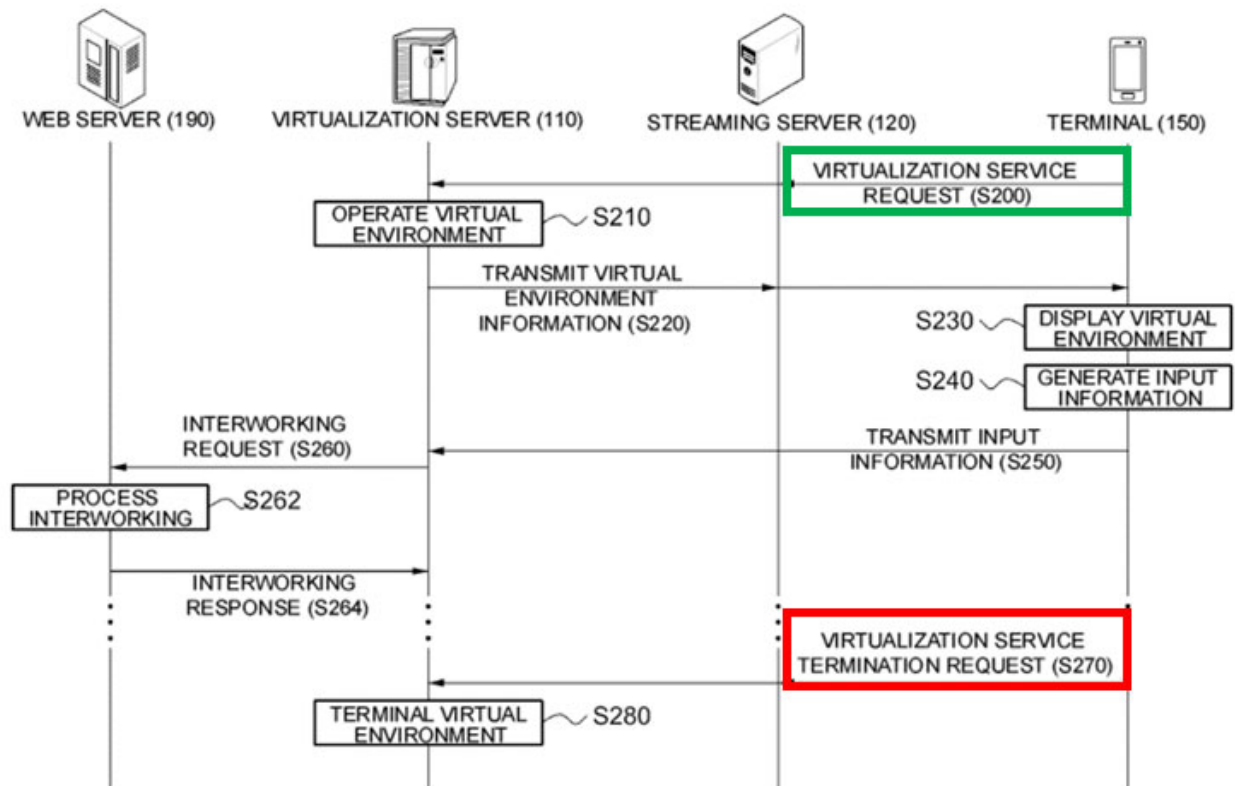
d. 28[c]

Yoon discloses or suggests 28[c] for the reasons discussed in Section VII.A.2.d. Martin, ¶100.

e. 28[d]

Yoon discloses or suggests 28[d]. Martin, ¶¶101-103.

Yoon discloses a terminal may initiate [green] and terminate [red] a virtualization service with the virtualization server. *E.g.*, Yoon, 17 (virtualization service request), 19 (virtualization service termination request), Fig. 2; Martin, ¶101.



Yoon, Fig. 2.

A POSITA would have understood a terminal can therefore engage in multiple virtualization sessions with the virtualization server. Martin, ¶102. Where a terminal engages in a second virtualization session with the virtualization server, a POSITA would have understood the virtualization server receives second input information from the mobile device. *Id.*

If PO argues this claim requires “different kinds of mobile device OS’s are executed and not merely multiple instances of the same mobile device OS” (as PO argued during prosecution, Section IV.B), Yoon likewise discloses or suggests this limitation. Martin, ¶103. In particular, Yoon discloses that “[a] plurality of mobile OS stack layers 430 may be provided” where “[e]ach of the plurality of mobile OS stack layers 430 may emulate a different OS.” Yoon, 22. A POSITA would have understood or found it obvious that Yoon’s virtualization server could execute multiple instances of the same mobile OS or different kinds of mobile OS’s. Martin, ¶103. This is consistent with Yoon’s stated objective of solving the problem where “an application that is developed for a specific smart phone OS may not be used for other smart phones that do not use the OS.” Yoon, 7. Emulating different kinds of mobile OS’s would allow a user to access applications that would not be available if only multiple instances of the same mobile OS were emulated. Martin, ¶103.

f. 28[e]

Yoon discloses or suggests 28[e]. Martin, ¶104.

As discussed in Section VII.A.8.e, Yoon discloses or suggests a terminal engaging in multiple virtualization sessions. Martin, ¶104. Yoon also discloses or suggests this limitation if the claim requires “different kinds of mobile OS’s are executed” for the reasons discussed in Section VII.A.8.e. *Id.* Yoon thus discloses 28[e] for the same reasons discussed in Section VII.A.2.c (“*processing...*” limitation) and because a second virtualization session would either be processed by “*a different instance than the first mobile device OS*” because the first instance would have been terminated already or be an entirely different kind of mobile OS. Martin, ¶104.

g. 28[f]

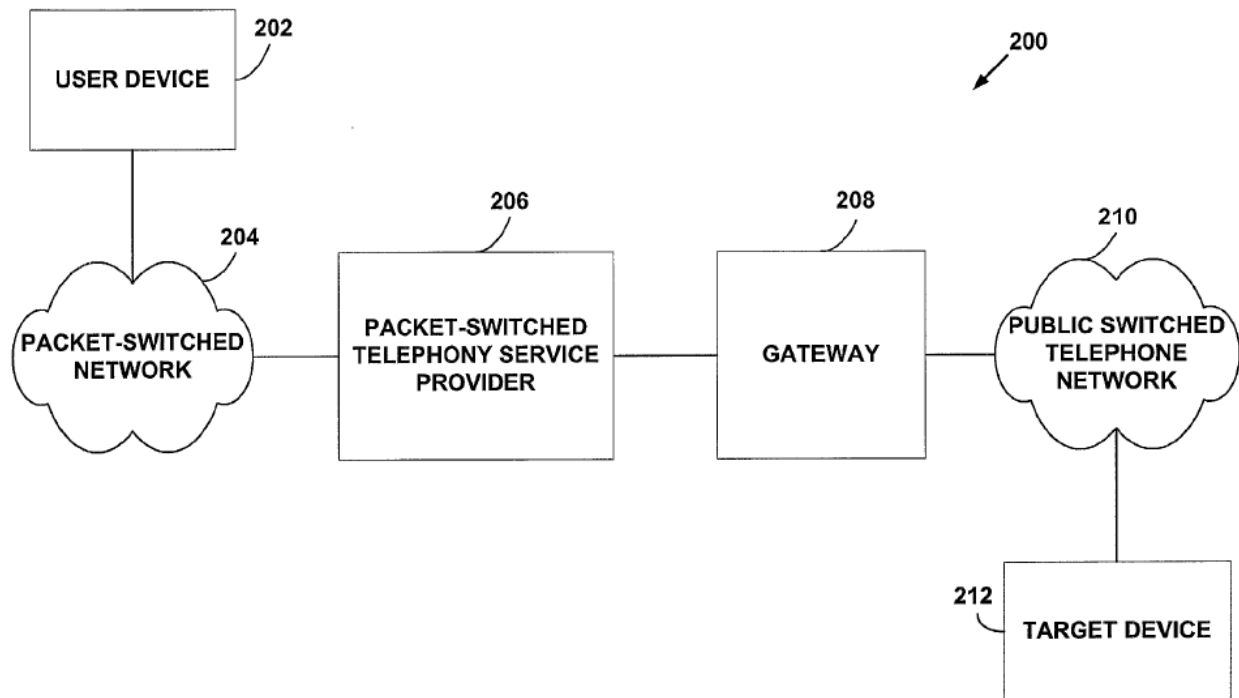
Yoon discloses or suggests 28[f] for the reasons discussed in Section VII.A.2.d. Martin, ¶105.

B. Ground 1B: Yoon, Nix (claims 4, 14, 22)

1. Nix

Nix teaches a “method for providing packet-switched telephony service.” Nix, Abstract. More specifically, Nix teaches a user device linked to an internet telephony service provider via a patch-switched network such as the internet and a gateway to a public switched telephone network (PSTN) linked to the internet telephony service provider on one end and a PSTN on the other, as illustrated below. Nix, ¶29, Fig. 2.

FIGURE 2



2. Motivation to Combine

Yoon discloses “various web services are provided by a web server.” Yoon, 7. Yoon further discloses the functions of its web server “may be integrated into the functions of the virtualization server 110.” Yoon, 16-17. In this petition, Petitioners map to an embodiment where the functions of Yoon’s web server are integrated into the virtualization server 110. Martin, ¶75. Yoon does not limit the types of web services that can be provided by the web server or provide any specific examples of web services. Martin, ¶107.

Nix discloses, “Internet Telephony,” which allows a device connected to the internet to “communicate through the Internet and PSTN [packet switched telephone

network] to a telephone subscriber at a telephone connected to the PSTN” with the aid of a gateway from the internet to the PSTN. Nix, ¶2. Nix explains “[n]umerous benefits may be realized through the use of packet-switched telephony,” including cheaper long-distance calls and “convenient interfaces and features [such as]...volume control, a video session, or an address book application.” Nix, ¶3. Nix further provides examples of “handheld device[s] or [] cellular phone[s]” as devices that would increasingly demand internet telephony capabilities. Nix, ¶7. A POSITA would have understood internet telephony to be a type of web service, as mentioned in Yoon. Martin, ¶¶108-109. By 2012 (when Yoon was published), internet telephony was widely available on mobile phones and smart phones, such as the terminals in Yoon. Martin, ¶109. A POSITA would have been motivated to provide a means for internet telephony on the emulated terminal/mobile OS/application because it would have been a common web service that provides users the benefits discussed by Nix. *Id.* Further, as mentioned by Yoon, providing access to internet telephony via Yoon’s web server would also allow access to this feature for user terminals that cannot themselves access it. *Id.*; Yoon, 11 (“Typically...the web service provided by the web server 190 is not able to interwork with an OS/application of the terminal.”). Doing so would have combined prior-art elements (Yoon’s web server for providing access to web services and Nix’s internet telephony, one such web service) according to known methods (providing a gateway

to a PSTN at a server, as disclosed by Nix) to yield predictable results (providing internet telephony to Yoon's mobile OS/application). Martin, ¶109. It also would have been no more than the use of a known technique (Nix's internet telephony provided at a server) to improve similar devices/methods (Yoon's web server) in the same way (allowing users to make telephone calls via the internet). *Id.*

3. Claim 4

The combination of Yoon and Nix renders obvious this claim. Martin, ¶¶110-112.

Yoon discloses the functions of a web server—which provides “various web services” that “may directly provide data through a web browser or a dedicated application”—are integrated into the functions of the virtualization server. Yoon, 7, 16-17. Accordingly, a POSITA would have understood Yoon's virtualization server provides “*a mobile device gateway for handling and providing internet access.*” Martin, ¶110.

Nix teaches providing a gateway to a PSTN at a packet-switched telephony service provider. Nix, Fig. 2. Voice data is transmitted between a user device and the packet-switched telephony service provider over a packet-switched network such as the internet. Nix, ¶33. The voice data is then transmitted to the gateway to the PSTN, again as packet-switched data. Nix, ¶33. The gateway to the PSTN transmits the voice data to a target device as circuit-switched data. Nix, ¶33. A POSITA would

have understood Nix teaches “*a mobile device gateway*” (the packet-switched telephony service provider) “*that handles voice calls, and a gateway to a PSTN.*” Martin, ¶111.

A POSITA would have been motivated to provide the function of the packet-switched telephony service provider described in Nix to the web server (and thus the virtualization server) of Yoon for the reasons discussed in Section VII.B.2. Martin, ¶112. In this combination, where Yoon’s web server (and thus virtualization server) provides internet telephony, among other web services, the virtualization server also provides “*voice calls and a gateway to a PSTN.*” *Id.*

4. Claim 14

The combination of Yoon and Nix renders obvious this claim for the reasons discussed in Section VII.B.2. Martin, ¶113.

5. Claim 22

The combination of Yoon and Nix renders obvious this claim for the reasons discussed in Section VII.B.2. Martin, ¶114.

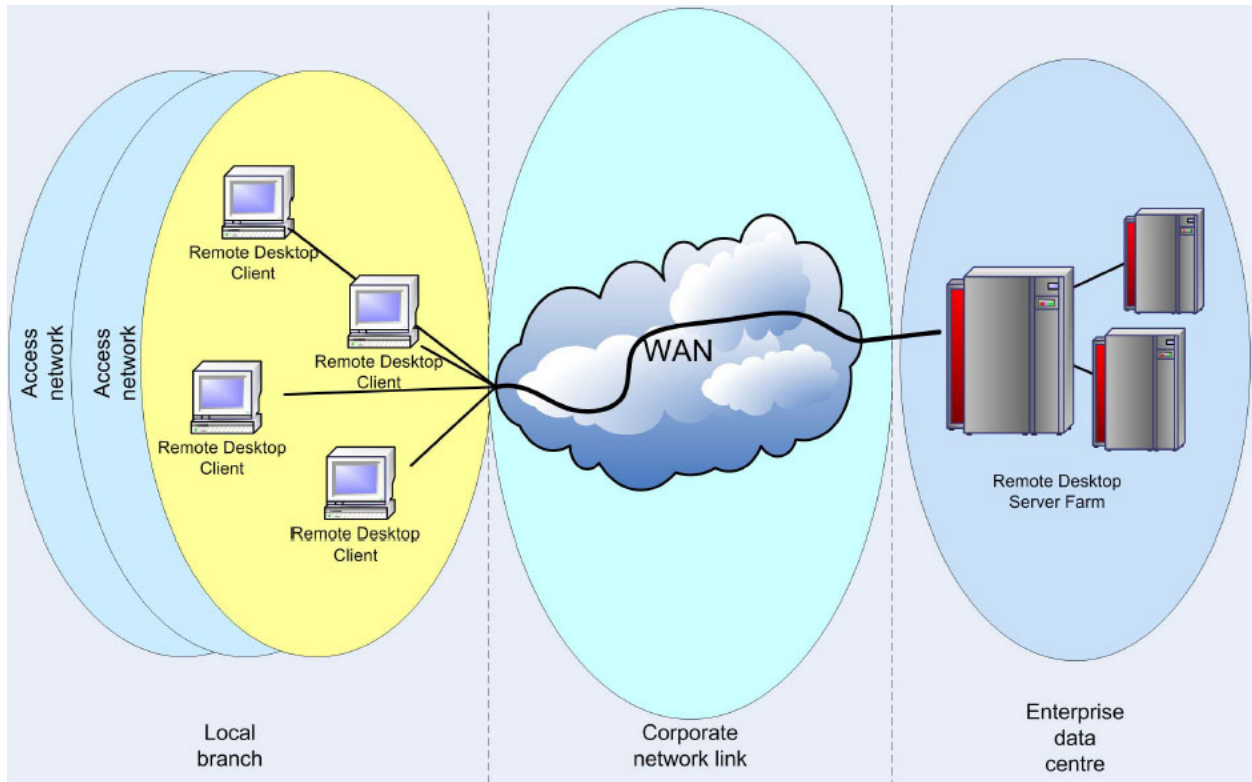
C. Ground 1C: Yoon, Lubonski, Richardson (claims 6-8, 11-13, 15-17, 23-25)

1. Lubonski

Lubonski teaches a “server-side component of the architecture for [a] remote desktop environment aiming at improving user perception of the more demanding and network sensitive multimedia applications and services.” Lubonski, Abstract.

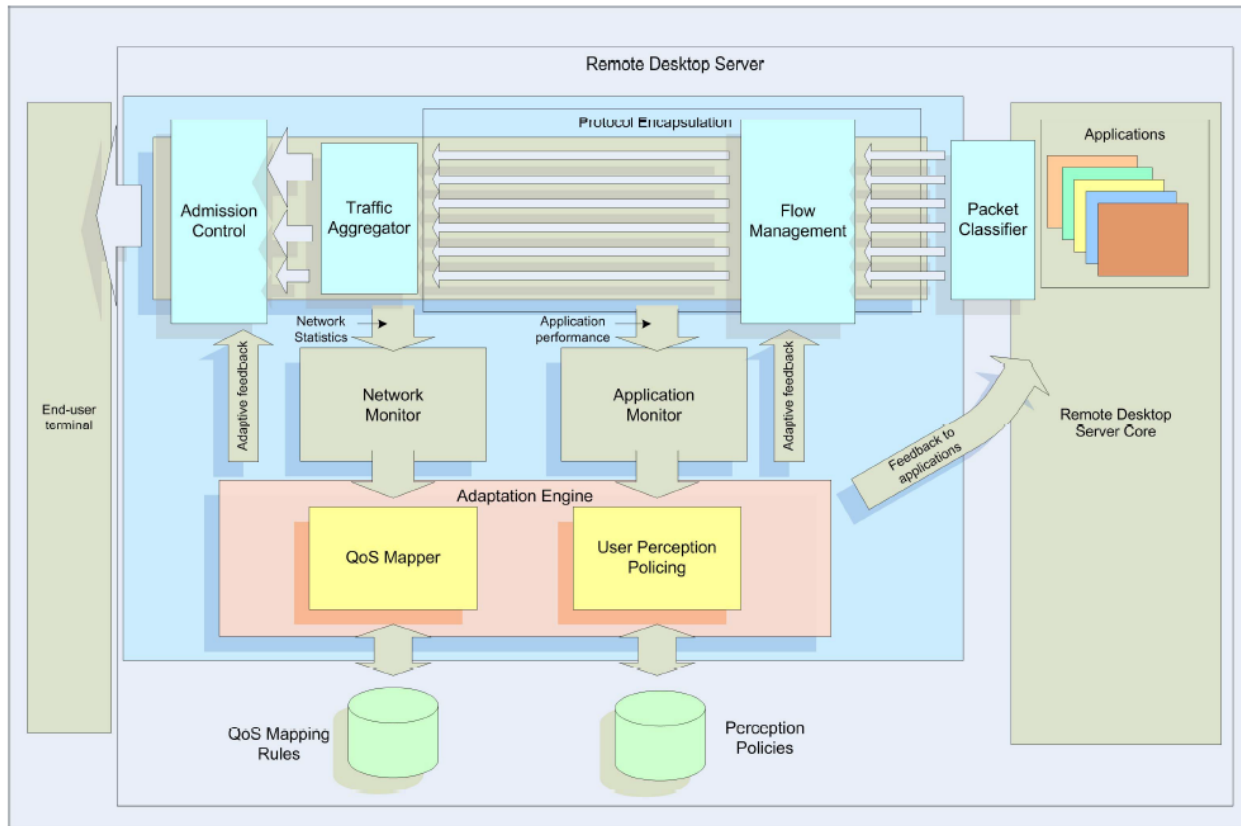
The “component maps [] user-perceived Quality of Service (QoS) onto network-level performance parameters in order to improve overall end-to-end QoS.” *Id.*

Lubonski is directed to a remote desktop environment where an end-user terminal connects, via a network, to a server. Lubonski, 152; Figure 1. “The effect of a virtual desktop is achieved by capturing screen updates by the server and encapsulation into remote desktop protocol. The protocol sends the desktop updates to the client and transmits back user input.” Lubonski, 152. Lubonski further explains “many different types of content, (such as textual data, multimedia streams or print spooling information)” are streamed between the server and user terminal. Lubonski, 152. “This leads to a more complex situation where[,] in the same connection from client to server[,] there is a multitude of flows, each with its specific network characteristics and requirements . . . [such as] time-insensitive data transfers such as WWW or FTP to the real-time transmissions of voice or video.” Lubonski, 152.



Lubonski, Figure 1.

Lubonski teaches “mapping of user-perception feedback to dynamically adapt resource parameters and[,] in result[,] improve the overall system performance.” Lubonski, 154. Lubonski accomplishes this with a server extension. Lubonski, 154, Figure 2:



Lubonski, Figure 2.

Lubonski’s extension “adjust[s] resource allocation in such a way that user-perceived QoS levels for all flows [between a server and end-user terminal] are preserved.” Lubonski, 155. To do this, Lubonski teaches a “Packet Classifier – responsible for the classification and marking of each application data flow – to reflect its traffic characteristics and user-perceived QoS requirements.” Lubonski, 155.

2. Richardson

Richardson is a textbook that describes itself as a “practical, accessible and unbiased guide to the H.264 video compression standard.” Richardson, Preface.

Relevant here, Richardson describes the transport of H.264 data. *Id.*, 244-248. Specifically, while H.264 itself “does not define a transport mechanism for coded visual data,” a well-known transport protocol used for H.264 data is Real-Time Protocol (“RTP”). *Id.*, 245-46. Richardson explains:

RTP defines a packet structure for real-time data (Figure 8.31) that includes a type identifier to signal the type of CODEC used to generate the data, a sequence number, essential for re-ordering packets that are received out of order, and a time stamp, necessary to determine the correct presentation time for the decoded data. Transporting a coded audio-visual stream via RTP involves packetizing each elementary stream into a series of RTP packets, interleaving these and transmitting them across an IP network.

Id., 246.

A simplified RTP-packet structure is provided in Figure 8.31:

	Payload Type	Sequence Number	Timestamp	Unique Identifier	Payload (e.g. Video Packet)
--	--------------	-----------------	-----------	-------------------	-----------------------------

Figure 8.31 RTP packet structure (simplified)

Richardson, Figure 8.31.

3. Motivation to Combine

Yoon's virtualization server encodes a "plurality of virtual environment images into a specific video format such as H.264 and may transmit the same to the terminal 150 in real time." Yoon, 15. Yoon's virtualization server also encodes a "plurality of time-based sounds into a specific video format such as H.264 and may transmit the same to the terminal 150 in real time." *Id.* While Yoon notes that transmitting images and sound as video may be more "efficient" (*id.*), Yoon does not provide further teaching regarding efficient data transmission between the virtualization server and terminals, nor does Yoon address the QoS of such transmissions. Martin, ¶122.

A POSITA would have understood QoS is important to user experience when streaming multimedia such as video and audio. Martin, ¶123 (discussing Rinne and Talaat). A POSITA would have further understood users value different QoS for different media types, depending on the application. Martin, ¶124 (citing Talaat). A POSITA implementing Yoon's teachings would therefore been motivated to improve QoS and would have looked to Lubonski for this improvement. Martin, ¶125.

Specifically, a POSITA would have been motivated to implement Lubonski's server extension that dynamically adapts resource parameters to maximize user-perceived QoS for multimedia transmitted from Yoon's virtualization server.

Martin, ¶126. A POSITA would have recognized Lubonski's extension, which improves user-perceived QoS for video and audio streaming between a server and client, would also improve Yoon's virtualization server and terminals connected over a network. *Id.* A POSITA would have also understood users perceive some changes in quality to be more significant than others (Lubonski, 155), and thus a POSITA would have been motivated to use Lubonski's extension to improve user experience for any given set of network resources by reserving resources, where possible, to QoS elements most valued by the user. Martin, ¶126.

To implement Lubonski's extension in Yoon's server, a POSITA would also have needed to understand how multimedia data is transported so that Lubonski's Packet Classifier could correctly distinguish between media types. Martin, ¶127. Because Yoon teaches encoding video and audio using H.264 (Yoon, 15) but provides no detail on how H.264 data is transported, a POSITA would have looked to an authoritative source like Richardson for that information. Martin, ¶127. In doing so, a POSITA would have seen RTP is one of two commonly-used transport protocols for H.264 data, and a POSITA would have been motivated to use RTP as one of a finite number of identified, predictable solutions, with a reasonable expectation of success. Richardson, 246; Martin, ¶127. When H.264 data is transmitted with RTP, each data packet includes a type identifier. Richardson, 246. A POSITA would have understood Lubonski's Packet Classifier would use this type

identifier to distinguish between, e.g., video and audio flows. Martin, ¶¶128-129; Lubonski, 155, Figure 2. A POSITA would have therefore been motivated to use RTP (as taught by Richardson) as the protocol for H.264 multimedia transport between Yoon's virtualization server—with Lubonski's extension for improving user-perceived QoS—and Yoon's terminals. Martin, ¶129. Doing so would have been no more than use of a known technique (RTP transport of H.264 data and Lubonski's server extension component) to improve a similar method (multimedia transport between a server and terminals) that was ready for improvement (Yoon's mention, without further teaching, of transmitting H.264 video between its virtualization server and user terminals) to yield predictable results (better user-perceived QoS). *Id.*

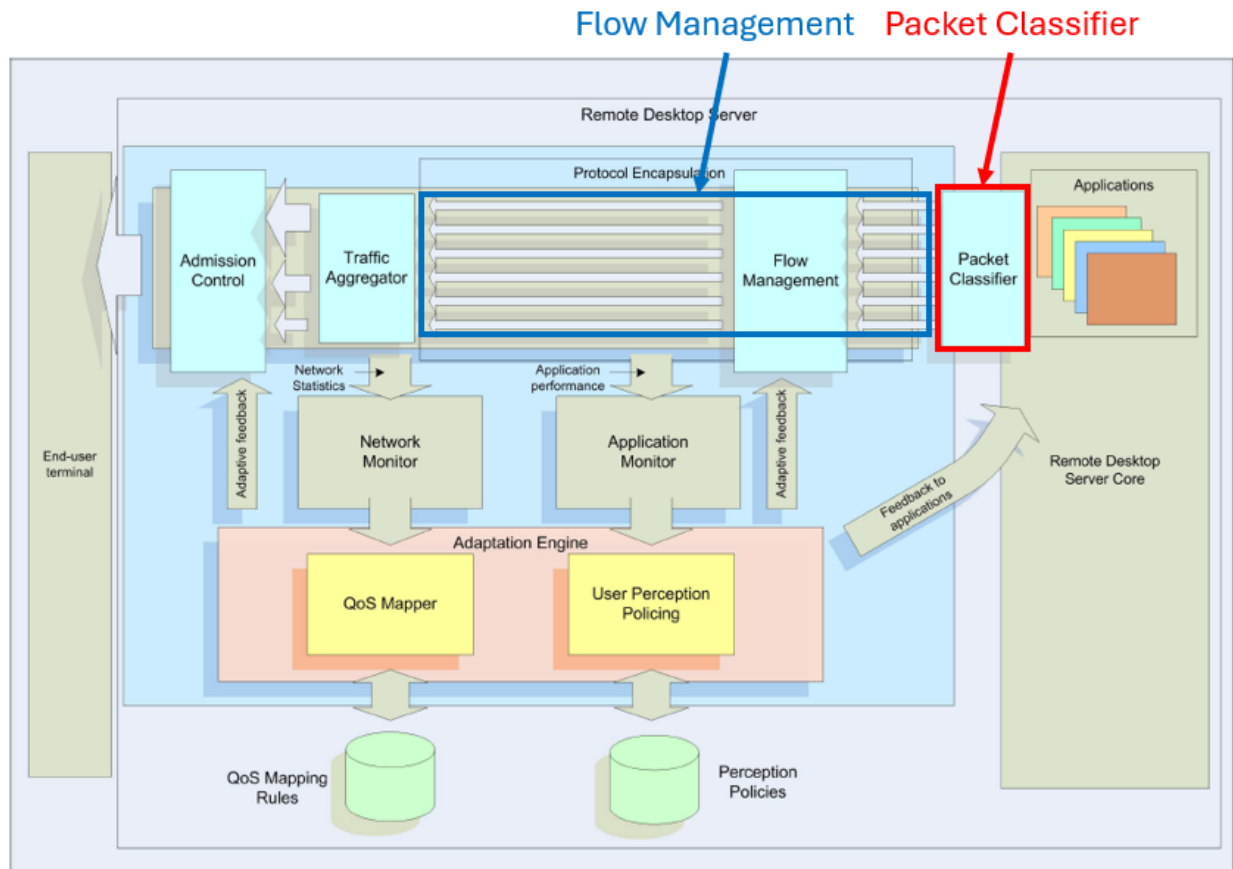
4. Claim 6

The Yoon-Lubonski-Richardson combination renders obvious this claim. Martin, ¶¶130-135.

Yoon discloses sending a plurality of different data types (e.g., images and audio) in a media stream from the virtualization server to terminal(s). Yoon, 14-15; Martin, ¶131.

Lubonski's server extension comprises a Packet Classifier [red] that "classif[ies] and mark[s] [] each application data flow - to reflect its traffic

characteristics and user-perceived QoS requirements.” [blue, comprising multiple data flows]. Lubonski, 155, Figure 2; Martin, ¶132.



Lubonski, Figure 2.

Lubonski’s extension then adjusts resource allocation among the data flows discerned by the Packet Classifier based on user-perceived QoS. Lubonski, 155; Martin, ¶133. Where RTP is used for multimedia transport (as explained in Richardson), the video and audio data produced by Yoon would be transported via packets indicating a different payload type. Richardson, 246, Figure 8.31.

Specifically, Richardson explains the RTP payload format “signal[s] the type of CODEC used to generate the data” and “RTP payload formats are defined for various standard video and audio CODECs, including H.264.” Richardson, 246; Martin, ¶134. A POSITA would have understood that H.264 does not provide for encoding audio, and thus a separate CODEC must be used to encode audio that accompanies video. Martin, ¶134. Accordingly, Lubonski's Packet Classifier would classify video and audio packets differently and separate them into different data flows so that user-perceived QoS of each data flow could be adjusted separately. *Id.* (Explaining users usually value video quality over audio quality and that, e.g., in such a case more resources would be devoted to the video flow at the expense of resources for the audio flow in order to improve user-perceived QoS). A POSITA would have understood each of these individually adjusted data flows for video and audio to be “*a separate data stream*” because each flow represents an independent sequence of data handled separately from the others. Martin, ¶135; Lubonski, 150 (“In our approach, the data flows of different characteristics-corresponding to different types of applications or services-within one client connection are being dynamically shaped to the current network conditions and user-perceived performance of remote applications and services.”).

5. Claim 7

The Yoon-Lubonski-Richardson combination renders obvious this claim. Martin, ¶136. Lubonski teaches adjusting resource allocation among data flows based on user-perceived QoS. Lubonski, 155. Lubonski further teaches, e.g., the QoS of video is particularly important to users. *Id.* Thus, a POSITA would have found it obvious to provide a different, higher, QoS for Yoon's video data flow than for Yoon's audio data flow. Martin, ¶136 (citing Talaat).

6. Claim 8

The Yoon-Lubonski-Richardson combination renders obvious this claim. Martin, ¶¶137-138. As explained in Sections VII.C.3 and VII.C.4, Yoon discloses encoding and transmitting video and audio from the virtualization server to terminal(s). While Yoon provides an example of encoding both images and sounds using the H.264 encoding standard (Yoon, 14-15), a POSITA would have understood H.264 does not provide an encoding method for audio, making it impossible to use the same encoding method for both video and audio. Martin, ¶137; Richardson, 246 (defined RTP payload formats for various standard video and audio CODECS). A POSITA would therefore have understood video and audio sent from Yoon's virtualization server would comprise different CODECs (and thus different encoding methods). Martin, ¶137.

Further, Lubonski teaches that adjusting parameters of a connection to improve user-perceived QoS comprises “allocating more bandwidth to [a] particular data flow if possible.” Lubonski, 155. And as explained in Section VII.C.5, a POSITA would have found it obvious to allocate more resources (and thus bandwidth) to the video data flow than the audio data flow. Martin, ¶138.

7. Claim 11

a. 11[pre]

Yoon discloses or suggests 11[pre] for the reasons discussed in Section VII.A.2.a. Martin, ¶139.

b. 11[a]

Yoon discloses or suggests 11[a] for the reasons discussed in Section VII.A.2.b. Martin, ¶140.

c. 11[b]

Yoon discloses or suggests 11[b] for the reasons discussed in Section VII.A.2.c. Martin, ¶141.

d. 11[c]

The Yoon-Lubonski-Richardson combination renders obvious 11[c]. Martin, ¶142. As discussed in Section VII.A.2.d, Yoon teaches “*sending, by the server, the stream of user interface information to the mobile device...to the mobile device...for reproduction....*” *Id.* As discussed in Section VII.C.4, the Yoon-Lubonski-

Richardson combination teaches the “*user interface information...comprises a plurality of different types of data and the server sends each different type of data to the mobile device in a separate data stream.*” *Id.*

8. Claim 12

The Yoon-Lubonski-Richardson combination renders obvious this claim for the reasons discussed in Section VII.A.3. Martin, ¶143.

9. Claims 13, 15-17, 23-25

The table below correlates the reasons why Yoon-Lubonski-Richardson teaches these claims. Martin, ¶¶144-151.

Claim	Substantially Similar Limitation/Claim	Reference
13[a]	3[a]	VII.A.4.a
13[b]	3[b]	VII.A.4.b
15	5	VII.A.5
16	7	VII.C.5
17	8	VII.C.6
23	6	VII.C.4
24	7	VII.C.5
25	8	VII.C.6

D. Ground 2A: Overton (claims 1-3, 5, 20-21, 28)

1. Overton

Overton teaches “emulating computing devices over a network” to allow “customers to use applications running remotely on emulated computing devices” because customers “may wish to try the application before buying it.” Overton, 1:54-65. As illustrated in the figure below, Overton teaches computing device 103 (e.g., a server) communicating with one or more clients 106. Overton, 2:24-36, Fig. 1 (below). Computing device 103 can execute emulation server application 115, and wrappers 118, applications 119, and media encoders 120. Overton, 2:57-63. “The emulation server application 115 is executed to launch applications 119, which are executed within the wrappers 118. Emulation server application 115 is also executed to obtain application input data 122 from the clients 106 and provide the application input data 122 to the respective wrapper 118.” Overton, 2:63-3:3. The emulation server application 115 also “send[s] application output data 123 that is captured from the application 119 to the clients 106.” Overton, 3:4-6.

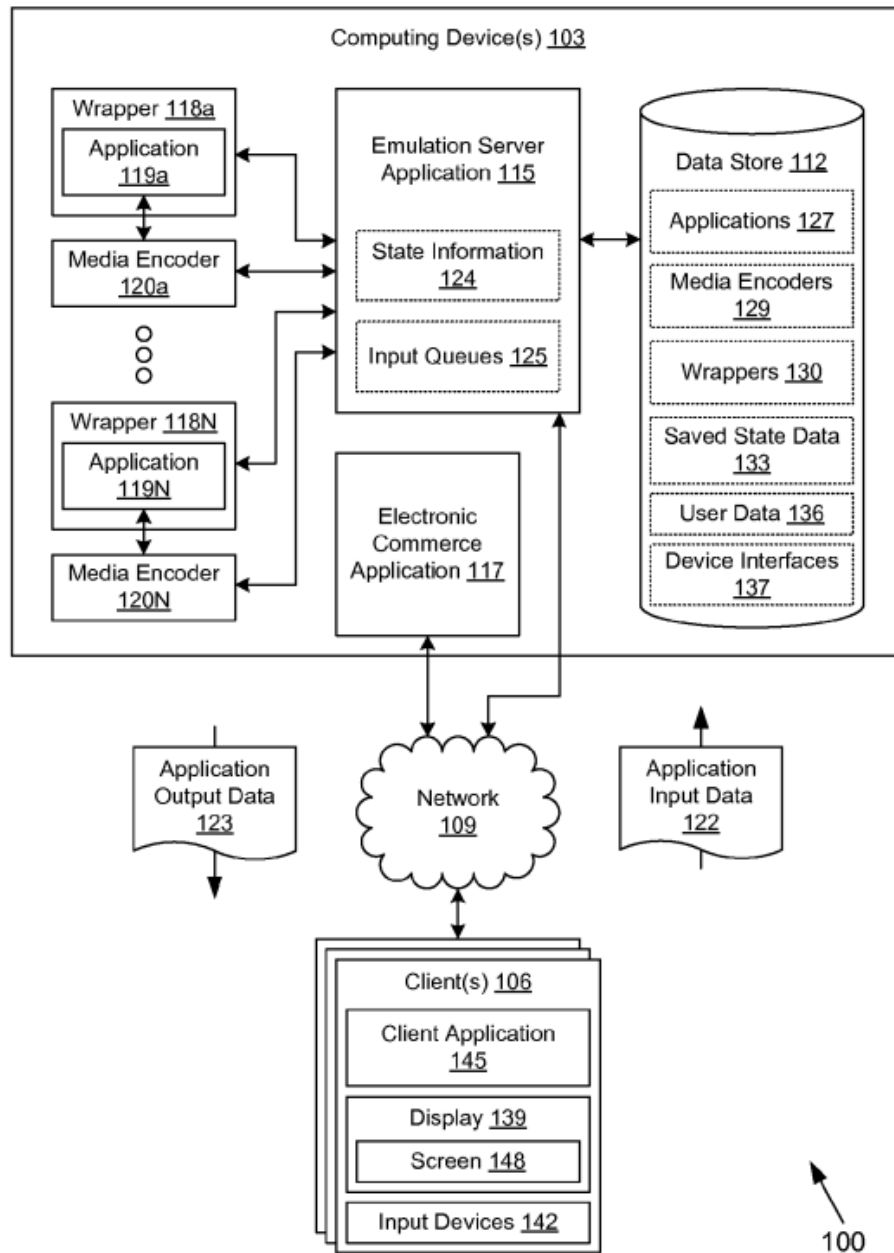


FIG. 1

Overton further discloses associated methods for emulating an application at its computing device 103. E.g., Overton, 11:52-12:46, Fig. 3.

2. Claim 1

a. 1[pre]

Overton discloses or suggests 1[pre]. Martin, ¶154.

Overton's computing device 103 ("*a server*") *emulates* a computing device at the request of a client 106 ("*a mobile device*"). Overton, 11:52-12:46, *see also, id.*, 2:34-36 ("The computing device 103 may comprise, for example, a server computer or any other system providing computing capability."), 4:45-49, Fig. 1 (below). Overton's mobile client devices include "a laptop computer, personal digital assistants, cellular telephones, [and] smartphones." Overton, 6:8-10; Martin, ¶154.

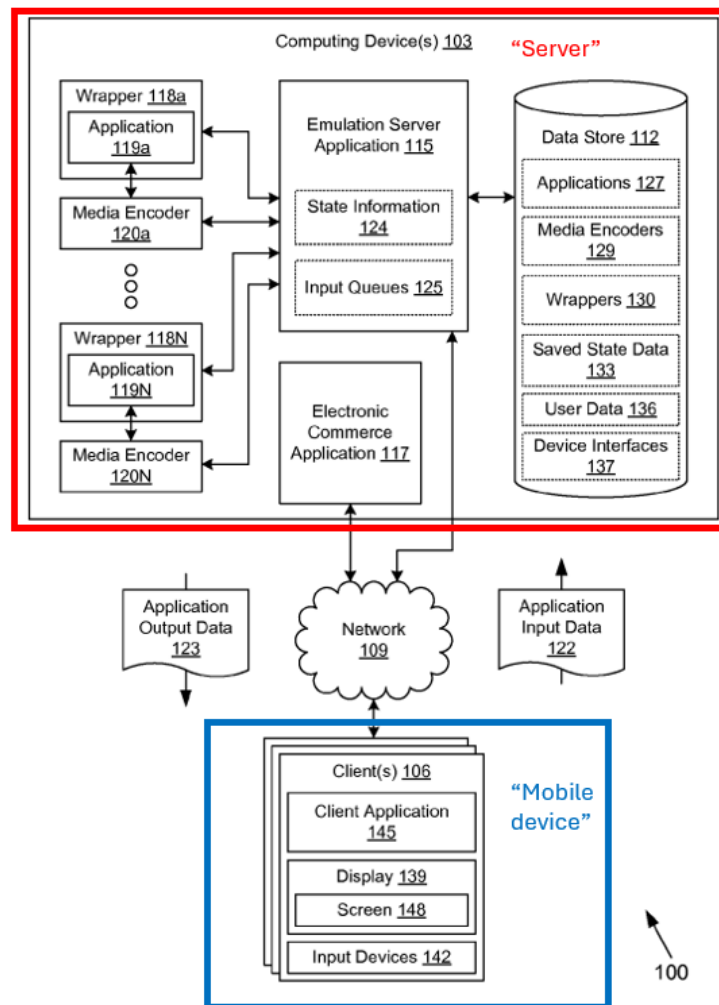


FIG. 1

b. 1[a]

Overton discloses or suggests 1[a]. Martin, ¶¶155-157.

As illustrated in Figure 1 below, Overton discloses client 106 [blue] executes client application 145 [teal] “configured to capture input commands provided by the user through one or more of the input devices 142 [purple] and send this input over the network 109 [yellow] to the computing device 103 [red] as application input data 122 [green].” Overton, 6:28-36. Computing device 103’s [red] (“the server”)

emulation server application 115 [brown] “obtain[s] application input data 122 from clients 106 and provide[s] the application input data 122 to the respective wrapper 118 [pink].” Overton, 2:67-3:3; Martin, ¶155.

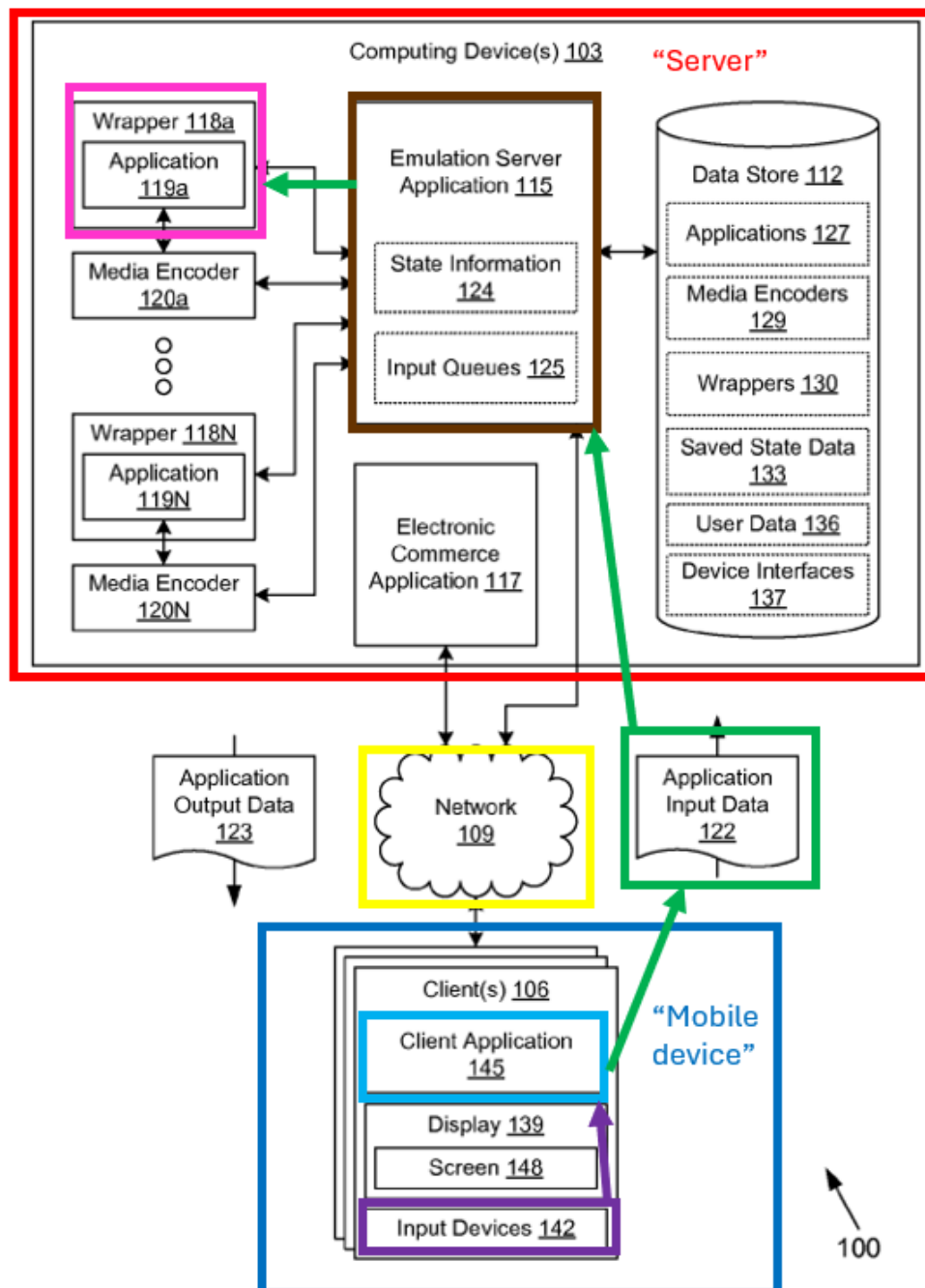


FIG. 1

Overton's client device 106 may be, e.g., a laptop, PDA, cell phone, or smart phone, each of which a POSITA would have understood to be "*a mobile device.*" "Overton, 6:4-14; Martin, ¶156. Thus, Overton discloses "*receiving, by the server, input information from at least one mobile device.*" Martin, ¶156.

Additionally, Overton discloses a sequence in which a user provides an input command at a client device, which then encodes⁵ the command and transmits it to the server, where the server processes the input command to perform a hardware action at the emulated mobile device. Overton, 8:64-9:7; *see also, id.*, 4:8-20. Because the user input is processed at the server to produce an action at the emulated application, a POSITA would have understood the user input is not processed by the client device to produce a display, audio, or HW action ("*the input information has not been processed to produce a display, audio and a hardware (HW) action by a HW component of the mobile device*"). Martin, ¶157.

c. 1[b]

Overton discloses or suggests 1[b]. Martin, ¶¶158-161.

⁵ Encoding does not encompass processing inputs "to produce a display, audio and a hardware (HW) action." Martin, ¶157, n.29, '887Pat, 7:38-50 (mobile device encodes inputs for transmittal to server).

Overton's application 119 executed at computing device 103 (*"the server"*) corresponds to an operating system and associated applications. Overton, 3:59-67. Application 119 is executed by wrapper 118 (Overton 2:65-67), which, as discussed in Section VII.D.2.b, receives application input data 122 and processes it. As explained in Section VII.D.2.a, Overton's server emulates a mobile device, so a POSITA would have understood Overton's *"mobile device operating system (OS) executed at the server"* processes the input information. Martin, ¶158.

As shown in Figure 1 below, Overton's application 119 generates a video and/or audio signal—i.e., application output data 123 [orange]. Overton, 3:4-6, 4:21-24. This application output data is sent by the server to client 106 (*"mobile device"*). Overton, 3:4-6. A POSITA would have understood this application output data is produced, at least in part, by processing the application input data received from the client device so that a user may try an application on the server before buying it—i.e., Overton's stated goal. Martin, ¶159; Overton, 1:54-58.

The video and/or audio generated at the server is encoded into a media stream (Overton, 4:28-32) and sent to client device 106 (*"mobile device"*) via a network (Overton, 6:41-44). The client device then renders the video on display 139 [black] and plays the audio (*"reproducing ... at least one of the display, the audio and the hardware (HW) action by the HW component of the mobile device"*). Overton, 6:41-46, Fig. 1; Martin, ¶160.

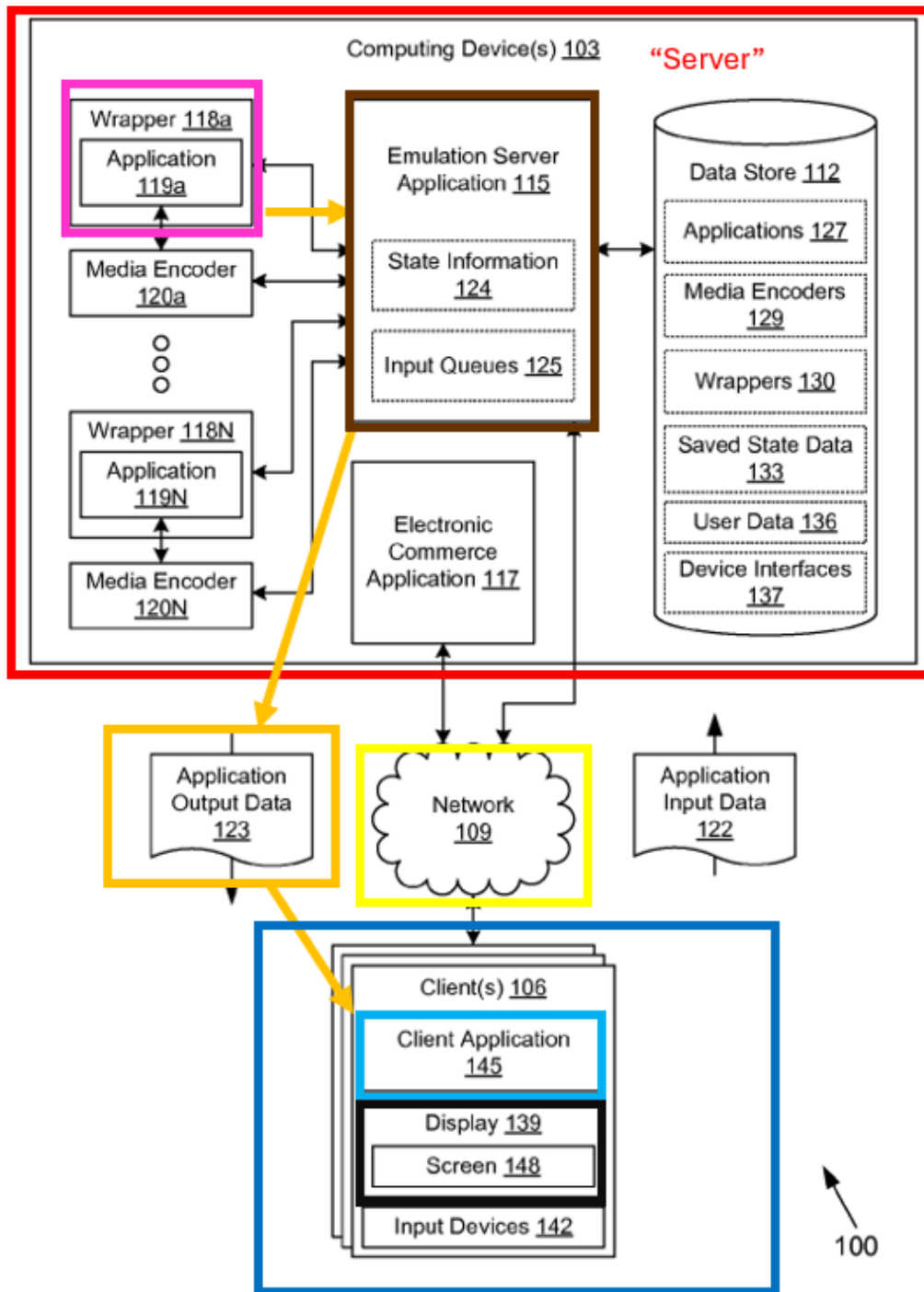


FIG. 1

Finally, because the same wrapper 118 and application 119 in Overton provide both the video and audio stream, as opposed to the video and audio stream

coming from different wrappers/applications and therefore different instances, a POSITA would understand that the streams are “*based on a same instance of the mobile device operating system.*” Martin, ¶161; ’887FH, 198.

d. 1[c]

Overton discloses or suggests 1[c]. Martin, ¶162.

Overton’s computing device 103 (“*the server*”) sends the media stream (“*user interface information*”) to a client device (“*mobile device*”). Overton, 4:21-39. The client device receives the media stream and renders video and audio from the stream on the device (“*reproduce at the mobile device at least one of the display, the audio and the HW action by the HW component of the mobile device.*”). Overton, 6:41-46; Martin, ¶162.

3. Claim 2

Overton discloses or suggests this claim. Martin, ¶163.

Overton’s server emulates mobile device hardware—e.g., “wrapper 118 may include an emulation application that emulates the device.” Overton, 4:46-49, 12:16-19 (“the emulation server application 115 loads the application 127 and executes an instance of an application 119 (FIG.1) within a wrapper 118 (FIG. 1) that corresponds to the emulated computing device.”); Martin, ¶163. As explained in Section VII.D.2.c, Overton teaches executing a mobile device OS in wrapper 118 (“*executing the mobile device on the emulated mobile device HW*”). Martin, ¶163.

4. Claim 3

a. 3[a]

Overton discloses or suggests 3[a]. Martin, ¶164.

As explained in Section VII.D.2.c, Overton’s computing device 103 (“*the server*”) executes a mobile device OS and associated applications. Overton, 3:59-67; Martin, ¶164.

b. 3[b]

Overton discloses or suggests 3[b]. Martin, ¶165.

As explained in Section VII.D.2.c, Overton’s application 119 executed by computing device 103 (“*the server*”) processes user input received from client device 106 (“*the input information*”) and “*produces the stream of user interface information.*” Martin, ¶165. If not explicitly disclosed, a POSITA would have found it obvious the application processes the input information to produce the stream of user interface information because Overton teaches allowing a user to try applications on the server before purchasing them (Overton, 1:56-59), which entails the user providing input to the application and receiving a stream of resulting user interface information. Martin, ¶165.

5. Claim 5

Overton discloses or suggests this claim. Martin, ¶166.

In Overton, “multiple clients 106 may be employed for one or more users to interact with the application 119.” Overton, 6:52-54. A POSITA would have understood application 119 at the computer device 106 (“*the server*”) receives user input from a plurality of different clients 106 (“*mobile devices*”). Martin, ¶166. A POSITA would have further understood the same application 119 (“*same instance of the mobile device OS*”) processes the input information received from the client devices (*see* Section VII.D.2.c) because Overton discloses computer device 103 may run multiple applications 119, but refers to a single application 119 as interacting with multiple clients. Overton, 6:52-54 (“multiple clients 106 may be employed for one or more users to interact with the application 119”); Martin, ¶166 (“interact” means each user sends input to and receives output from the application).

6. Claim 20

a. 20[pre]

Overton discloses or suggests 20[pre] for the same reasons discussed in Section VII.D.2.a. Martin, ¶167.

b. 20[a]

Overton discloses or suggests 20[a] for the same reasons discussed in Sections VII.D.2.b, VII.D.5. Martin, ¶168. Overton’s computing device 103 receives input information that has not been processed from a client device (Section VII.D.2.b),

and multiple client devices may interact with the same application 119 at the computer device (Section VII.D.5).

c. 20[b]

Overton discloses or suggests 20[b] for the same reasons discussed in Sections VII.D.2.c, VII.D.5. Martin, ¶169. Overton teaches the same application 119 processes input information from multiple clients 106 and produces a media stream to send to each of the clients, where the stream will be rendered. Sections VII.D.2.c, VII.D.5.

d. 20[c]

Overton discloses or suggests 20[c] for the same reasons discussed in Sections VII.D.2.d, VII.D.5 because Overton discloses sending the media stream from computer device 103 to one or more client devices 106 (“*at least one of the plurality of different mobile devices*”). Martin, ¶170.

7. Claim 21

a. 21[a]

Overton discloses or suggests 21[a] for the same reasons discussed in Section VII.D.4.a. Martin, ¶171.

b. 21[b]

Overton discloses or suggests 21[b] for the same reasons discussed in Section VII.D.4.b. Martin, ¶172.

8. Claim 28

a. 28[pre]

Overton discloses or suggests 28[pre] for the same reasons discussed in Section VII.D.2.a. Martin, ¶173.

b. 28[a]

Overton discloses or suggests 28[a]. Martin, ¶174.

As discussed in Section VII.D.2.b, Overton’s computer device 103 (“*server*”) receiving input information from client device 106 (“*mobile device*”). Overton receives multiple input commands (i.e., at least first and second inputs) from a client device. Overton, 6:32-36 (“the client application 145 is configured to capture input **commands** provided by the user through **one or more of the input devices 142** and send this input over the network 109 to the computing device 103 as application input data 122.”); Martin, ¶174.

c. 28[b]

Overton discloses or suggests 28[b] for the same reasons discussed in Section VII.D.2.c. Martin, ¶¶175-178.

Alternatively, if PO argues this claim requires “different kinds of mobile device OS’s are executed and not merely multiple instances of the same mobile device OS” (as PO argued during prosecution, Section IV.B), Overton discloses or suggests this limitation. Martin, ¶176. In Overton, a user may choose to emulate

different devices to test an application before purchasing it. For example, Overton's Figure 2A (below) depicts an application available for purchase that is compatible with two different devices:

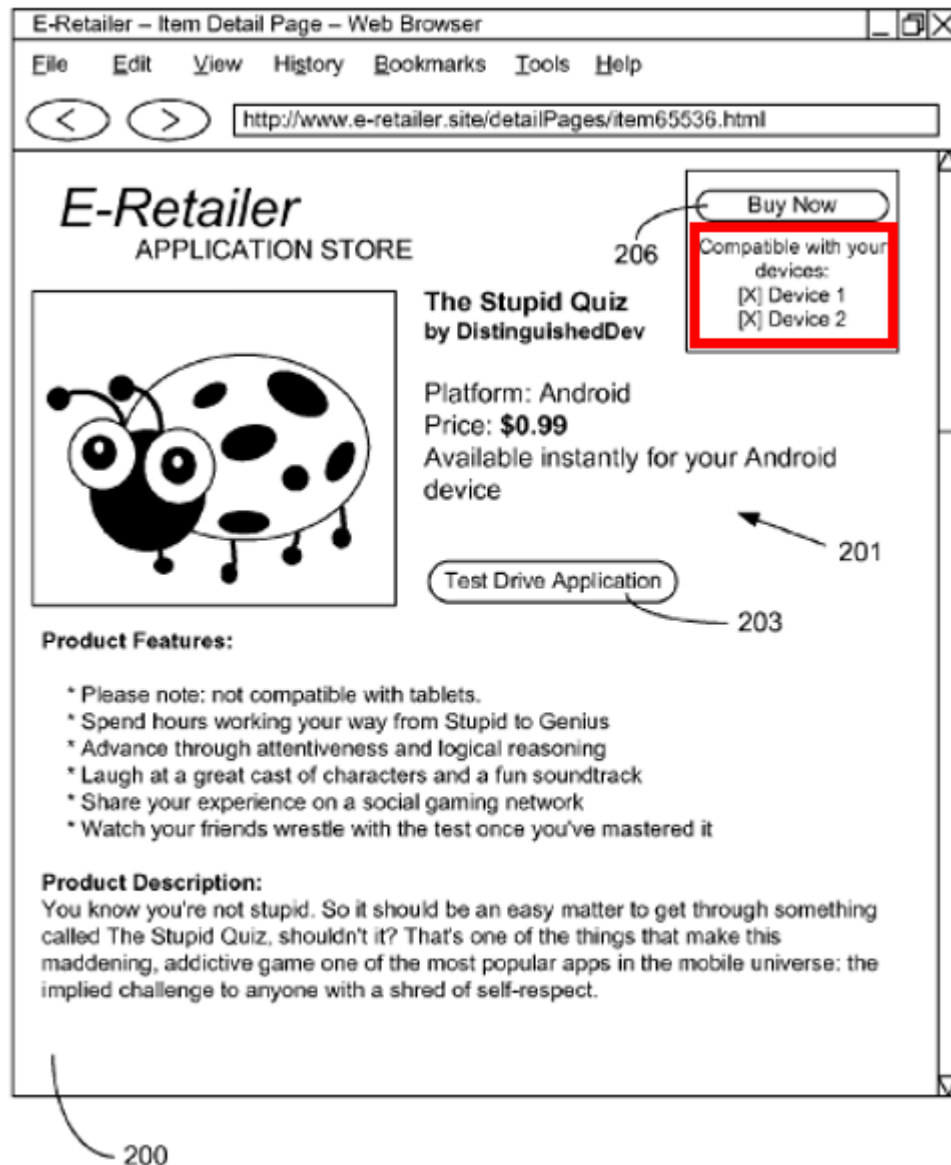


FIG. 2A

“[W]hen a customer activates the emulation request tool 203, the customer is able to use the application 127 in a virtualized environment that resembles the

computing device on which it is intended to execute” and provides the “non-limiting example [where] the customer is associated with two computing devices that are compatible with the application.” Overton, 7:11-22. “The customer may specify one or more preferred models of an emulated computing device for which the customer is seeking applications,” and the server emulates a device “specified by the customer at the client 106.” Overton, 8:31-33, 11:66-12:12. Overton’s applications executed by computing device 103 may have been originally designed for different types of devices, such as, e.g., a general-purpose computer, smartphone, or a handheld game device. Overton, 3:55-59; Martin, ¶177.

A POSITA would have understood different types of devices may have different kinds of mobile OS’s. Martin, ¶178. Accordingly, a POSITA would have understood Overton teaches allowing a user to emulate different kinds of mobile OS’s to try an application on different kinds of devices. *Id.* For example, a user can try an application on “device 1” as depicted in Figure 2A and afterwards try the application on “device 2,” where these are two different kinds of devices with different kinds of mobile OS’s emulated by Overton’s server. *Id.* Overton thus discloses or suggests operating a first kind of mobile OS followed by a second kind of mobile OS at its server. *Id.*

d. 28[c]

Overton discloses or suggests 28[c] for the same reasons discussed in Section VII.D.2.d. Martin, ¶179.

e. 28[d]

Overton discloses or suggests 28[d] for the same reasons discussed in Sections VII.A.2.b (the server receiving an input), VII.D.8.b (the user selecting to emulate two different devices, with different kinds of mobile OS's, one after the other). Martin, ¶180.

f. 28[e]

Overton discloses or suggests 28[e]. Martin, ¶¶181-184.

Overton teaches client device 106 interacting with multiple (i.e., first and second) applications (“*mobile device operating systems*”) at computing device 103, and Overton explains users can try applications executed on the server before purchasing them. Overton, 1:55-58. Overton further teaches multiple different applications 119 are executed by computer device 103. Overton, 2:57-63, Fig. 1. A POSITA would have found it obvious that a user shopping for applications may wish to try out multiple applications before choosing to purchase one (or more). Martin, ¶181.

Overton further teaches each application emulation session is eventually terminated, either by a timer, user choice, or inactivity. Overton, 12:32-39; Fig. 3.

Accordingly, a POSITA would have understood when a user tries multiple applications, each application would be tried in its own emulation session that is terminated by the user before trying out the next because Figure 3 does not provide for trying multiple applications in the same emulation session. Martin, ¶182. Accordingly, the second application tried by the user would be in a “*different OS instance.*” Martin, ¶182.

If PO argues this claims requires “different kinds of mobile device OS’s are executed and not merely multiple instances of the same mobile device OS” as it did during prosecution, Overton discloses or suggests this limitation as discussed in Section VII.D.8.c. Martin, ¶183.

Overton discloses the remainder of 28[e] for the reasons discussed in Section VII.D.2.c. Martin, ¶184.

g. 28[f]

Overton discloses or suggests 28[f] for the same reasons discussed in Section VII.D.2.d. Martin, ¶185.

E. Ground 2B: Overton, Nix (claims 4, 14, 22)

1. Motivation to Combine

A POSITA would have been motivated to connect a gateway to a PSTN, as taught by Nix, to Overton’s computing device 103, acting as a server, to allow users

to make voice calls and/or try applications for internet telephony. Martin, ¶¶186-190.

Nix explains “[n]umerous benefits may be realized through the use of packet-switched telephony,” including cheaper long-distance calls and “convenient interfaces and features [such as]...volume control, a video session, or an address book application.” Nix, ¶3. Nix further discloses several known applications that provide packet-switched telephony services. Nix, ¶¶3, 4; Martin, ¶187.

Overton explains a goal of its invention is to allow a customer to try an application before purchasing it. Overton, 1:56-58. A POSITA would have understood users may want to try different applications for packet-switched telephony and would have been motivated to combine the teachings of Nix and Overton to allow users to try them. Martin, ¶188. Specifically, because Overton already teaches a user device communicating with a computing device (server) over a network such as the internet (Overton, 2:24-32, Fig. 1), a POSITA would have been motivated to connect its computing device to a gateway to a PSTN, as taught by Nix, to allow users to try applications for packet-switched telephony. Martin, ¶188. Doing so would have been no more than combining prior art elements (Overton’s computing device acting as a server and communicating with a mobile device over a network, and Nix’s server communicating with a mobile device over a network and a gateway to a PSTN) according to known methods (providing

internet telephony via a server communicating with a gateway to a PSTN) to yield predictable results (allowing users to try internet telephony applications before purchasing them). Martin, ¶188.

Alternatively, a POSITA would have understood that in cases where client devices 106 are cellular telephones or smartphones (together, “phones”), Overton, 6:4-10, a core functionality of phones is to allow the user to place telephone calls. Martin, ¶189. Because Overton’s Figure 1 shows that client devices are only connected to computing device 103, a POSITA would have been motivated to modify computing device 103 to provide this core functionality for phones that connect to it. Martin, ¶189

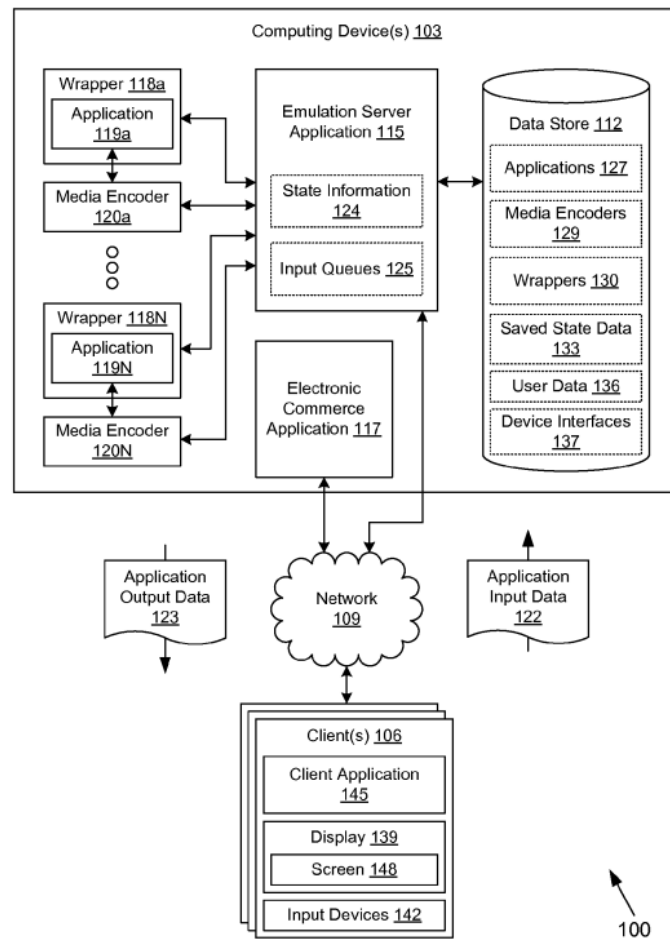


FIG. 1

Overton, Fig. 1.

A POSITA would have therefore connected a gateway to a PSTN, as taught by Nix, to Overton's computing device 103, to provide telephone functionality to phones that connect to computing device 103. Martin, ¶190. Doing so would have amounted to nothing more than combining prior art elements (Nix's gateway to a PSTN and Overton's computing device 103 connected via a network to client devices 106) according to known methods (connecting a gateway to a PSTN to a

server, as taught in Nix) to yield predictable results (providing the ability for Overton's phone client devices to place telephone calls). *Id.*

2. Claim 4

The Overton-Nix combination renders obvious this claim. Martin, ¶¶191-195.

Overton teaches allowing users to try applications before purchasing them. Overton, 1:56-58. A POSITA would have understood that at least some such applications (e.g., internet browser applications) provide internet access to the user. Martin, ¶191. A POSITA would have thus understood Overton's computing device 103 is a "*mobile device gateway for handling and providing internet access*" by virtue of allowing the user to try applications that provide internet access. *Id.*

Nix teaches providing a gateway to a PSTN at a packet-switched telephony service provider. Nix, Fig. 2. Voice data is transmitted between a user device and the packet-switched telephony service provider over a packet-switched network such as the internet. Nix, ¶33. The voice data is then transmitted to the gateway to the PSTN, again as packet-switched data. Nix, ¶33. The gateway to the PSTN transmits the voice data to a target device as circuit-switched data. Nix, ¶33. A POSITA would have understood Nix teaches a mobile device gateway (the packet-switched telephony service provider) that handles voice calls, and a gateway to a PSTN. Martin, ¶192.

In combining the teachings of Overton and Nix, a POSITA would have been motivated to incorporate Nix's teaching of a packet-switched telephony service provider into one of the applications 119 taught by Overton because, as is disclosed by Nix and was also well-known at the time, various applications that provided packet-switched telephony existed (e.g, Skype or Viber). Martin, ¶193; Nix, ¶3. A POSITA would have further been motivated to provide, at the computing device 103 ("*server*"), a gateway to a PSTN, as taught by Nix, to allow users to try these telephony applications. Martin, ¶193.

Alternatively, a POSITA would have been motivated to provide the core functionality of making telephone calls to Overton's client devices that are phones, and would have thus connected Nix's gateway to a PSTN to Overton's computing device 103 for the reasons discussed in Section VII.E.1. Martin, ¶194.

Accordingly, Overton and Nix teach providing, by Overton's computing device 103 ("*server*") applications for internet access and packet-switched telephony ("*mobile device gateway for handling and providing internet access, voice calls*") and a gateway to a PSTN to allow the applications for packet-switched telephony to place voice calls. Martin, ¶195.

3. Claim 14

The Overton-Nix combination renders obvious this claim for the reasons discussed in Section VII.E.2. Martin, ¶196.

4. Claim 22

The Overton-Nix combination renders obvious this claim for the reasons discussed in Section VII.E.2. Martin, ¶197.

F. Ground 2C: Overton, Lubonski, Richardson (claims 6-8, 11-13, 15-17, 23-25)

1. Motivation to Combine

A POSITA would have been motivated to improve user-perceived QoS for media transmitted from Overton's computing device 103 to client device(s) 106 by implementing the teachings of Lubonski's server extension and Richardson's use of RTP for transmitting H.264 data. Martin, ¶¶198-203.

Overton discloses various encoders for the video and/or audio stream sent from its computing device 103 to client device(s) 106. Overton, 4:28-39. Overton explains "[s]uch media encoders 120 may be selected according to factors such as, for example, data reduction, encoding quality, latency, etc." Overton, 4:34-36. Overton thus recognizes quality concerns associated with transmitting a media stream and suggests varying encoders to deal with quality issues. Martin, ¶199. Overton further teaches its computing device 103 may include, e.g., the H.264 encoder (Overton, 4:31-34), and computing device 103 and client 106 may communicate using RTP (Overton, 3:6-11).

A POSITA would have understood that, in addition to selecting a suitable encoder, adjusting the available resources to improve user perceived QoS (as taught

by Lubonski) would improve the user experience. Martin, ¶200. A POSITA would have done so by implementing Overton's teachings because, e.g., a poor user experience when trying an application might make the user less likely to purchase the application. Martin, ¶200; *see also*, Section VII.C.3 (discussing importance of QoS to user experience). A POSITA implementing Overton's teachings would have thus found it worthwhile to improve QoS from the user's point of view, and in doing so, a POSITA would have looked to the teachings of Lubonski. Martin, ¶200.

Specifically, a POSITA would have been motivated to implement Lubonski's server extension that dynamically adapts resource parameters to maximize user-perceived QoS for multimedia sent from Overton's computing device. Martin, ¶201. A POSITA would have recognized Lubonski's extension, which improves user-perceived QoS for video and audio streaming between a server and client connected by a network, would work on Overton's computing device 103 and client(s) 106 that are connected by a network. Martin, ¶201. A POSITA would have also understood users perceive some changes in quality to be more significant than others (Lubonski, 155) and therefore would have been motivated to use Lubonski's extension to improve a user's experience for any given set of network resources. Martin, ¶201.

While Overton discloses encoding multimedia via H.264 and transmitting data via RTP, a POSITA would have looked to Richardson for additional detail to implement the encoding and transport. In doing so, a POSITA would have seen that

when H.264 data is transmitted with RTP, each data packet includes a type identifier. A POSITA would have also understood that H.264 does not provide for encoding audio, and thus a separate CODEC (identified by a different payload type in RTP packets) must be used to encode audio that accompanies video. Martin, ¶202; Richardson, 246 (defined RTP payload formats for various standard video and audio CODECS). A POSITA would have understood Lubonski's Packet Classifier would use the type identifier to distinguish between, e.g., video and audio flows. Martin, ¶202; Lubonski, 155, Fig. 2.

A POSITA would have thus been motivated to use RTP, as suggested by Overton and taught by Richardson, as the protocol for multimedia transport between Overton's computing device 103, with Lubonski's component for improving user user-perceived QoS, and Overton's client device(s). Martin, ¶203. Doing so would have been no more than use of a known technique (RTP transport of H.264 data and Lubonski's server extension component) to improve similar methods (multimedia transport between Overton's computing device and client device(s)) using a known approach that was ready for improvement (Overton's brief references to H.264 encoding and RTP transport) to yield predictable results (improved user-perceived QoS). *Id.*

2. Claim 6

The Overton-Lubonski-Richardson combination renders obvious this claim. Martin, ¶¶204-206.

Overton discloses sending a plurality of different types of data (e.g., video and audio) in a media stream from the computing device 103 (server) to client device(s) 106. Overton, 10:38-44; Martin, ¶205. Overton suggests audio data “may be captured and multiplexed into the media stream.” Overton, 10:40-41.

A POSITA looking to improve Overton’s QoS would have looked to Lubonski and Richardson’s teachings for the reasons explained in Section VII.F.1. Lubonski’s server extension comprises a Packet Classifier that classifies and marks each application data flow. Lubonski, 155, Fig. 2. The extension then adjusts resource allocation among the data flows discerned by the Packet Classifier based on user-perceived QoS. Lubonski, 155. Where RTP is used for multimedia transport (as suggested by Overton and described by Richardson), Overton’s video and audio data (“*the stream of user interface information comprises a plurality of different types of data*”) would be transported via packets indicating different payload types. Richardson, 246, Fig. 8.31. Specifically, Richardson explains the RTP payload format “signal[s] the type of CODEC used to generate the data” and “RTP payload formats are defined for various standard video and audio CODECs, including H.264.” Richardson, 246. Accordingly, in the combination, Lubonski’s Packet

Classifier would classify video and audio packets differently and separate them into different data flows so that user-perceived QoS of each data flow could be adjusted separately. Martin, ¶206. A POSITA would have understood each of these individually adjusted data flows for video and audio to be “*a separate data stream*” because each flow represents an independent sequence of data handled separately from the others. Martin, ¶206; Lubonski, 150 (“In our approach, the data flows of different characteristics-corresponding to different types of applications or services-within one client connection are being dynamically shaped to the current network conditions and user-perceived performance of remote applications and services.”).

3. Claim 7

The Overton-Lubonski-Richardson combination renders obvious this claim. Martin, ¶207. Lubonski teaches adjusting resource allocation among data flows based on user-perceived QoS. Lubonski, 155. Lubonski further teaches, e.g., the QoS of video is particularly important to users, even more so than the QoS of audio. *Id.* So a POSITA would have found it obvious to provide a *different QoS* for Overton’s video data flow (“*at least one of said separate data streams*”) as compared to Overton’s audio data flow (“*another of said separate data streams*”). Martin, ¶207.

4. Claim 8

The Overton-Lubonski-Richardson combination renders obvious this claim. Martin, ¶¶208-210.

As explained in Sections VII.F.1, VII.F.2, Overton encodes and transmits video and audio from computing device 103 to client device(s). While Overton provides an example of encoding H.264 video, a POSITA would have known H.264 does not provide an encoding method for audio. Martin, ¶209; Richardson, 246. Thus, a POSITA would have understood Overton’s video and audio sent from the computing device (“*separate data streams*”) would comprise different CODECs (and thus “*different ... encoding methods*”). Martin, ¶209.

Further, Lubonski teaches adjusting parameters of a connection to improve user-perceived QoS comprises “allocating more bandwidth to [a] particular data flow if possible.” Lubonski, 155. As explained in Section VII.F.3, a POSITA would have found it obvious to allocate resources differently (and thus bandwidth) for the video data flow and the audio data flow. Martin, ¶210.

5. Claim 11

a. 11[pre]

Overton discloses or suggests 11[pre] for the same reasons discussed in Section VII.D.2.a. Martin, ¶211.

b. 11[a]

Overton discloses or suggests 11[a] for the reasons discussed in Section VII.D.2.b. Martin, ¶212.

c. 11[b]

Overton discloses or suggests 11[b] for the reasons discussed in Section VII.D.2.c. Martin, ¶213.

d. 11[c]

The Overton-Lubonski-Richardson combination renders obvious 11[c]. Martin, ¶214. As discussed in Section VII.D.2.d, Overton discloses “*sending, by the server, the stream of user interface information to the mobile device...to the mobile device...for reproduction....*” *Id.* As discussed in Section VII.F.2, the Overton-Lubonski-Richardson combination teaches that the “*user interface information...comprises a plurality of different types of data and the server sends each different type of data to the mobile device in a separate data stream.*” *Id.*

6. Claim 12

The Overton-Lubonski-Richardson combination renders obvious this claim for the reasons discussed in Section VII.D.3. Martin, ¶215.

7. Claims 13, 15-17, 23-25

The table below correlates the reasons why Yoon-Lubonski-Richardson teaches these claims. Martin, ¶¶216-223.

Claim	Substantially Similar Limitation/Claim	Reference
13[a]	3[a]	VII.D.4.a
13[b]	3[b]	VII.D.4.b

15	5	VII.D.5
16	7	VII.F.3
17	8	VII.F.4
23	6	VII.F.2
24	7	VII.F.3
25	8	VII.F.4

G. Ground 3A: Lu (claims 1-3, 5, 20-21)

1. Lu

Lu discloses methods and apparatuses to “start a game appliance for a client system” such that “said client system can send input to the game appliance system via a network to control game play of the game appliance and in return receive media stream of the game appliance via a network.” Lu, Abstract, Claim 1. Lu explains a game appliance “comprises a package of at least one game software application and/or game software appliance” that “includes but [*sic*] not limited to, computer game, mobile game, console game, online game client, virtual world client, video game, etc.” Lu, ¶¶19, 22. The game appliance “can also comprise [an] emulation environment required by a game application (e.g., console emulation environment, handset emulation environment, portal gaming emulation environment, etc.).” Lu, ¶21. Lu’s game appliance system comprises a game appliance apparatus that can

execute a game appliance by “emulat[ing] a target platform or a target operating system along with the game appliance running over the target platform or the target operating system.” Lu, ¶45, *see also, id.*, ¶¶43-44, 99, 113, 120. Lu’s game appliance apparatus provides “interactive game play” for a client system by using “input data sent from a client system.” Lu, ¶¶67-70. Specifically, a “client system can send input to the game appliance apparatus via network to control game play of the game appliance and in return receive media stream of the game appliance via a local network.” Lu, ¶87. Figure 1 (below) depicts its game appliance system [red], comprising the game appliance apparatus [orange], and a client system [blue].

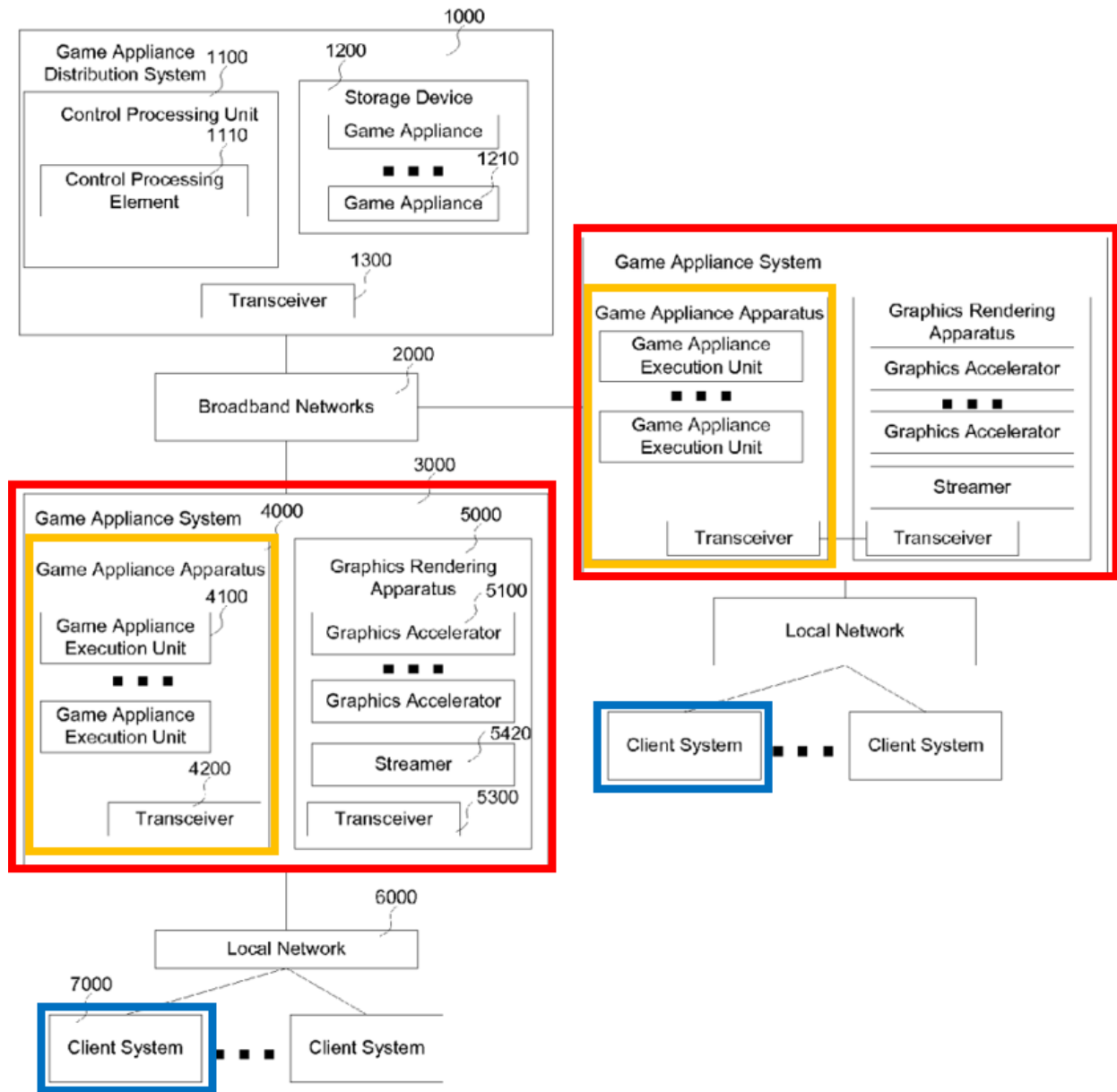


Fig. 1

Lu, Fig. 1.

2. Claim 1

a. 1[pre]

Lu discloses or suggests 1[pre]. Martin, ¶226.

Lu discloses its game appliance system (“*a server*”) comprises a game appliance apparatus that can use emulation to execute a game appliance “originally developed for a different operating system.” Lu, ¶¶43, 46 (game appliance apparatus can emulate a target platform), 99, 113, 120. Lu further discloses “the kind of game application contained in a game appliance includes...[a] mobile game.” Lu, ¶22. Because Lu discloses emulating a target platform for a game appliance, and a game appliance can include a mobile game, a POSITA would have understood Lu emulates a device used to play a mobile game, i.e., a “*mobile device*.” Martin, ¶226; *see also*, Lu, ¶46 (game appliance apparatus can run a Gameboy emulator or a Smartphone emulator).

b. 1[a]

Lu discloses or suggests 1[a]. Martin, ¶¶227-228.

Lu discloses a client device (“*mobile device*”) sends an input to the game appliance apparatus within the game appliance system (“*the server*”), and the game appliance apparatus uses the input data sent from the client system to control a game appliance it executed. Lu, ¶¶69-70. A POSITA would have understood the game appliance apparatus (and thus, “*the server*”) *receives* the input from the client device. Martin, ¶227. Because Lu discloses the client system sends input and receives a media stream in return, e.g., Lu, ¶87, a POSITA would have understood “*the input information has not been processed to produce a display, audio and a hardware*

(HW) action by a HW component of the mobile device.” Martin, ¶227. The only input processing Lu contemplates may be performed at the client system is “translation” of the input, but Lu teaches translation could occur either at the client or at the game appliance apparatus (*“the server”*), specifically providing an example where “a client system can transmit the original input event. Then, the game appliance apparatus that receives the event does any required translation.” Lu, ¶77; Martin, ¶228.

c. 1[b]

Lu discloses or suggests 1[b]. Martin, ¶¶229-234.

In addition to emulating a mobile device, as discussed in Section VII.G.2.a, Lu discloses its game appliance apparatus emulates a *“mobile device operating system (OS).”* Lu, ¶¶20 (game appliance comprises operating system for running game applications), 43 (game appliance apparatus can use operating system emulation to execute a game appliance), 45 (game appliance apparatus can emulate a target operating system). Because Lu’s game appliance apparatus executes mobile games, a POSITA would have understood the game appliance apparatus also executes *“a mobile device operating system (OS).”* Martin, ¶229.

Lu further discloses input sent from the client system to the game appliance apparatus is used “to control game play of the game appliance and in return receive [sic] media stream of the game appliance.” Lu, ¶87. Thus, Lu’s game appliance (and

the mobile device OS executed at the game appliance apparatus) processes the input to create “*a stream of user interface information.*” Martin, ¶230. Lu’s “client system...supports rendering of the media streams transmitted to it from a game appliance apparatus or a graphics rendering apparatus over a local network. It can decompress the received media stream and display the frames on an attached display device.” Lu, ¶61. Lu thus discloses the media stream is used in “*reproducing at the mobile device at least one of the display, the audio and the hardware (HW) action by the HW component of the mobile device.*”⁶ Martin, ¶230.

If PO argues—as the Applicant argued during prosecution—this limitation requires multiple data streams be provided by a single emulation of the mobile device, Lu renders obvious this limitation. Martin, ¶¶232-233; ’887FH, 198 (“Put differently, applicants’ claimed invention provides multiple data streams (e.g., audio, video, compressed text, voice and/or sensors) arriving from a single mobile

⁶ Lu teaches that “[t]he actual graphics rendering can be carried out by a game appliance apparatus or a graphics rendering apparatus,” Lu, ¶105, so a POSITA would have understood Lu discloses embodiments where graphics rendering is performed by the same game appliance apparatus that receives input from the client devices, i.e., “*based on a same instance of the mobile device operating system.*” Martin, ¶231.

device emulation.”). While Lu focuses on the graphics rendering aspect of executing a game application at the game appliance apparatus, a POSITA would have understood (or at least found it obvious) Lu also renders audio associated with game play and to include that audio in the media stream from the game appliance apparatus to the client system. Martin, ¶232. Specifically, a POSITA would have known most games include audio and would have understood the game appliance apparatus generates that audio when executing the game. *Id.* Indeed, Lu does not disclose that game applications are modified in any way to exclude audio, nor does it disclose rendering audio separately, at the client system (or how one would do so), so a POSITA would have understood in executing game applications, both video and audio would be produced by the game appliance apparatus. *Id.*

Alternatively, a POSITA would have been motivated to configure Lu to perform audio processing at the game appliance apparatus and to include audio in the media stream to the client devices. Martin, ¶233. Doing so would have simplified the system by requiring only one instance of the game application to be run, at the game appliance apparatus, to provide both video and audio associated with gameplay. *Id.* This configuration also would have obviated the need to somehow modify a gaming application to render video at one device (the game appliance apparatus) and associated audio at a separate device (the client system). *Id.* (explaining that a POSITA would not have known how to make such a

modification). This is consistent with Lu's disclosure, which does not provide any teaching for executing any part of the game application at the client system (or any other way of obtaining audio associated with a game application). *Id.* It further would have been obvious to try as one of two possible locations, the client system or the game appliance system, for rendering audio, and a POSITA would have had a reasonable expectation of success. *Id.*

Furthermore, as mentioned above, Lu's game appliance apparatus alone executes a game application. *See also*, Lu, ¶99 (single game appliance execution unit with game appliance apparatus can execute a game appliance using emulation). So if this limitation requires multiple data streams (e.g., video and audio) to be provided from a single emulation, Lu discloses or renders obvious this limitation. Martin, ¶234.

d. 1[c]

Lu discloses or suggests 1[c]. Martin, ¶235.

Lu's appliance apparatus (part of the game appliance system ("*the server*")) sends a media stream back to the client system. Lu, Abstract, ¶¶49, 61, 64, 81, 87. The client system (which can be a "mobile device such as [a] PDA or cellphone or handset, netbook, laptop computer, etc) supports rendering of the media streams transmitted to it from a game appliance apparatus." Lu, ¶61; Martin, ¶235. The client

system “decompress[es] the received media stream and display[s] the frames on an attached display device.” Lu, ¶¶61, 64, 66.

3. Claim 2

Lu discloses or suggests this claim. Martin, ¶236.

Lu’s game appliance apparatus can emulate an entire target platform to execute a game application. Lu, ¶¶43-46. A POSITA would have understood emulating a target platform to include emulating the target platform hardware and executing the associated OS on the emulated hardware. Martin, ¶236, *see also*, Lu, ¶70 (disclosing emulating input device hardware).

4. Claim 3

a. 3[a]

Lu discloses or suggests 3[a]. Martin, ¶237. Lu’s game appliance apparatus (part of the game appliance system (“*the server*”)) executes a game application. Lu, ¶¶30, 43-46.

b. 3[b]

Lu discloses or suggests 3[b]. Martin, ¶238. Lu’s game appliance (which comprises a game “*application*”) receives “*input information*” from the client system. Lu, ¶¶69-72, 99. The executed game appliance then provides a graphics rendering command (Lu, ¶99) to create the media stream sent back to the client

system (“*the stream of user interface information*”). Lu, Abstract, ¶¶49, 61, 64, 81, 87; Martin, ¶238.

5. Claim 5

Lu discloses or suggests this claim. Martin, ¶¶239-240. Lu discloses “one...game appliance apparatus[] . . . can retrieve game appliance and execute game appliance for one or a collection of client systems.” Lu, ¶26. Thus, a POSITA would have understood Lu discloses an embodiment where a single game appliance apparatus (and thus a single “*mobile device OS*”) processes input from multiple different client systems (“*mobile devices*”) to produce the stream of user interface information that goes back to those systems. Martin, ¶239. Such an embodiment also would have been obvious because Lu discloses emulating gaming systems that were well-known for providing local multiplayer games. Lu, ¶46; Martin, ¶240. Because Lu discloses multiple client systems connecting to a single emulated game appliance, a POSITA would have understood the server receives input information from a plurality of different mobile devices, acting as controllers for multiple players, engaged in local multiplayer play at the emulated game appliance. Martin, ¶240.

6. Claim 20

a. 20[pre]

Lu discloses or suggests 20[pre] for the reasons discussed in Section VII.G.2.a. Martin, ¶241.

b. 20[a]

Lu discloses or suggests 20[a] for the reasons discussed in Sections VII.G.2.b (receiving input information that has not been processed) and VII.G.5 (a single game appliance apparatus executes game appliances for, and thus receives input from, multiple client systems). Martin, ¶242.

c. 20[b]

Lu discloses or suggests 20[b] for the reasons discussed in Sections VII.G.2.c (“*processing...*” for a single client system) and VII.G.5 (multiple client systems to a single game appliance apparatus). Martin, ¶243.

d. 20[c]

Lu discloses or suggests 20[c] for the reasons discussed in Sections VII.G.2.d (“*sending...*”) and VII.G.5 (multiple client systems to a single game appliance apparatus). Martin, ¶244.

7. Claim 21

a. 21[a]

Lu discloses or suggests 21[a] for the reasons discussed in Section VII.G.4.a. Martin, ¶245.

b. 21[b]

Lu discloses or suggests 21[b] for the reasons discussed in Section VII.G.4.b. Martin, ¶246.

H. Ground 3B: Lu, Lubonski, Richardson (claims 6-8, 11-13, 15-17, 23-25)

1. Motivation to Combine

A POSITA would have been motivated to improve user-perceived QoS for media transmitted from Lu's game appliance apparatus within the game appliance system ("*the server*") to client systems by implementing the teachings of Lubonski's server extension and Richardson's use of RTP for transmitting H.264 data. Martin, ¶247.

Lu recognizes the need for, and discloses some methods for ensuring, QoS for the media stream to the client system. Martin, ¶248. For example, Lu discloses "specif[ying] a desired rendering frame rate or frame rendering period." Lu, ¶160, *see also*, Lu, ¶¶182, 183 ("transcoding can also be performed according to achievable or pre-determined bitrate. In an embodiment, bitrate can be determined based on the available or assigned bandwidth between a client system and a game appliance system."). Thus, Lu teaches maintaining a QoS by encoding the media stream according to the available bandwidth between the client system and game appliance system. Martin, ¶249. However, a POSITA would have recognized bandwidth varies throughout the duration of a network connection, and thus would have looked for additional solutions to maximize QoS, e.g., in cases where available bandwidth changes. *Id.*

In a similar field, Lubonski suggests improving user-perceived QoS for media streamed between a server and client with a server-side component that adjusts resource allocation among packet flows from a server to a client to preserve user-perceived QoS. Lubonski, 155; Martin, ¶249. For example, Lubonski notes a decrease in displayed frames per second, a parameter Lu suggests adjusting to meet QoS requirements, can create a “significant deterioration in user-perceived performance.” Lubonski, 155, Lu, ¶160. A POSITA would have thus recognized Lu’s approach to maintaining QoS may have shortcomings if it disproportionately affected user-perceived QoS. Martin, ¶249. Recognizing the importance of the user experience, particularly in gaming applications (to which Lu is directed), a POSITA would have been motivated to implement Lubonski’s approach to improving user-perceived QoS for a better user experience. *Id.* (discussing importance of low latency video for gaming applications). Doing so would have required no more than applying a known technique (Lubonski’s technique for improving user-perceived QoS) to a known device ready for improvement (Lu’s game appliance system that provides limited teachings for maintaining QoS) to yield predictable results (improved user-perceived QoS given the same network resources). *Id.*

Because Lubonski teaches using a packet classifier to mark data flows sent from the server (Lubonski, 155, Fig. 2), a POSITA would have needed to understand the structure of packets sent from Lu’s game appliance system to a client system.

Martin, ¶250. Lu discloses the game appliance system encoding video using H.264, Lu, ¶56, and using RTP to stream media to a client system, Lu, ¶187. A POSITA would have thus looked to a resource on H.264 such as Richardson for details related to the transmission of H.264 encoded media using RTP. Richardson teaches that RTP packets include an identifier of the type of CODEC, e.g., H.264 for video and an audio CODEC for accompanying audio, used to generate the data in the packet payload. Richardson, 246; Martin, ¶250. A POSITA would have therefore had a reasonable expectation of success in combining Lubonski's teachings with Lu's because Lubonski's packet classifier would use the type identifier in the RTP packets already used by Lu to transmit the media stream to the client system. *Id.*

2. Claim 6

The Lu-Lubonski-Richardson combination renders obvious this claim. Martin, ¶¶251-253.

As explained in Section VII.G.2.c, it would have been obvious to include both video and audio data in the media stream sent from Lu's game appliance system to a client system. Martin, ¶252.

A POSITA looking to improve Lu's quality of service would have looked to Lubonski's teachings for the reasons explained in Section VII.H.1. Lubonski's server extension comprises a Packet Classifier that classifies and marks each application data flow. Lubonski, 155, Fig. 2. The extension then adjusts resource

allocation among the data flows discerned by the Packet Classifier based on user-perceived QoS. Lubonski, 155. Because Lu discloses sending the media stream to the client system via RTP (which includes a type identifier with each packet), Lubonski's Packet Classifier would classify video and audio packets differently and separate them into different data flows so that user-perceived QoS of each data flow could be adjusted separately. Martin, ¶253. A POSITA would have understood each of these individually adjusted data flows for video and audio to be a “*separate data stream*” because each flow represents an independent sequence of data handled separately from the others. Martin, ¶253; Lubonski, 150 (“In our approach, the data flows of different characteristics - corresponding to different types of applications or services - within one client connection are being dynamically shaped to the current network conditions and user-perceived performance of remote applications and services.”).

3. Claim 7

The Lu-Lubonski-Richardson combination renders obvious this claim. Martin, ¶254. Lubonski teaches adjusting resource allocation among data flows based on user-perceived QoS. Lubonski, 155. Lubonski further teaches, e.g., the QoS of video is particularly important to users, even more so than the QoS of audio. *Id.* A POSITA would have found it obvious to provide a different QoS for Lu's video

data flow (*“at least one of said separate data streams”*) as compared to Lu’s audio data flow (*“another of said separate data streams”*). Martin, ¶254.

4. Claim 8

The Lu-Lubonski-Richardson combination renders obvious this claim. Martin, ¶¶255-257.

Lu discloses video may be encoded using any video or image compression approach, e.g., H.264. Lu, ¶¶56, 182. Lu does not disclose a specific encoding method for audio, but a POSITA would have understood audio would need to be encoded for efficient transmission. Martin, ¶256. A POSITA would have further understood none of the video or image encoding approaches Lu discloses can be used to encode audio, so the POSITA would have used a different encoding method, e.g., AAC (Advanced Audio Codec). Martin, ¶256.

Further, Lubonski teaches adjusting parameters of a connection to improve user-perceived QoS comprises “allocating more bandwidth to [a] particular data flow if possible.” Lubonski, 155. As explained in Section VII.H.3, a POSITA would have found it obvious to allocate resources differently (and thus bandwidth) for the video data flow and the audio data flow. Martin, ¶257.

5. Claim 11

a. 11[pre]

Lu discloses or suggests 11[pre] for the reasons discussed in Section VII.G.2.a. Martin, ¶258.

b. 11[a]

Lu discloses or suggests 11[a] for the reasons discussed in Section VII.G.2.b. Martin, ¶259.

c. 11[b]

Lu discloses or suggests 11[b] for the reasons discussed in Section VII.G.2.c. Martin, ¶260.

d. 11[c]

The Lu-Lubonski-Richardson combination renders obvious 11[c] for the reasons discussed in Sections VII.G.2.d (“*sending, by the server, the stream of user interface information to the mobile device...to the mobile device...for reproduction...*”), VII.H.2 (“*user interface information...comprises a plurality of different types of data and the server sends each different type of data to the mobile device in a separate data stream*”). Martin, ¶261.

6. Claim 12

The Lu-Lubonski-Richardson combination renders obvious this claim for the reasons discussed in Section VII.G.3. Martin, ¶262.

7. Claims 13, 15-17, 23-25

The table below correlates the reasons why Yoon-Lubonski-Richardson teaches these claims. Martin, ¶¶263-270.

Claim	Substantially Similar Limitation/Claim	Reference
13[a]	3[a]	VII.G.4.a
13[b]	3[b]	VII.G.4.b
15	5	VII.G.5
16	7	VII.H.3
17	8	VII.H.4
23	6	VII.H.2
24	7	VII.H.3
25	8	VII.H.4

VIII. DISCRETIONARY DENIAL FACTORS FAVOR INSTITUTION

The weight of the factors articulated in *Apple Inc. v. Fintiv, Inc.*, IPR2020-00019, Paper 11, 6 (Mar. 20, 2020) (precedential) favors institution.

Factor 1 is neutral. Petitioners have not requested a stay, but intend to do so.

Factor 2 favors institution. Trial is currently scheduled for April 6, 2026, around the same time as the expected FWD. But the Director’s Interim Procedure for Discretionary Denials notes that “[a] court’s scheduled trial date...is not by itself

a good indicator of whether the district court trial will occur before the statutory deadline.” The latest Federal Court Management statistics, EX1014, state that the median time to trial for the Northern District of California is 48.9 months. This places trial in July 2028, over two years after the expected FWD date.

Factor 3 strongly favors institution. Petitioners filed this petition at an early stage in the litigation, just over four months after being served with the Complaint, nearly two months before their invalidity contentions are due, and nearly 7 months before the scheduled *Markman* hearing.

Factor 4 favors institution because Petitioners challenge far more claims than are at issue in the district court. EX1013 (alleging infringement of claims 11, 15-17, 20, 23-25).

Factor 5 favors institution because the parties are the same and the FWD is expected before trial, such that estoppel will apply under 35 U.S.C. 315(e).

Factor 6 strongly favors institution because the merits here are compelling.

IX. CONCLUSION

Inter Partes Review and cancellation of the Challenged Claims is respectfully requested.

X. MANDATORY NOTICES

A. Real Party in Interest

Petitioner identifies themselves as real parties in interest.

B. Related Matters

To the best of Petitioner's knowledge, the '887 Patent has been involved in the following matters:

- *BSD Crown, Ltd. f/k/a Emblaze, Ltd. v. Amazon.com Inc.*, NDCA-3-24-cv-03503

C. Notice of Counsel and Service Information

LEAD COUNSEL
Jessica Kaiser (Reg. No. 58,937) kaiser-ptab@perkinscoie.com Perkins Coie LLP 1900 Sixteenth Street, Suite 1400 Denver, CO 80202-5255 Telephone: (303) 291-2300
BACK-UP COUNSEL
Jon R. Carter, Reg. No. 75,145 PERKINS COIE LLP 1155 Avenue of the Americas 22nd Floor New York, NY 10036-2711 Tel: (212) 262-6900 carter-ptab@perkinscoie.com Atanas Baitchev, Reg. No. 82,414 PERKINS COIE LLP 11452 El Camino Real Suite 300 San Diego, CA 92130-2080

Tel: (858) 720-5700
baitchev_ptab@perkinscoie.com

Respectfully submitted,

/ Jessica Kaiser /

Jessica Kaiser
Reg. No. 58,937
Attorney for Petitioner

PERKINS COIE LLP
1900 Sixteenth Street, Suite 1400
Denver, CO 80202-5255

Date: October 24, 2024

CERTIFICATE OF WORD COUNT UNDER 37 CFR §42.24(d)

Pursuant to 37 C.F.R. §42.24(a), Petitioner hereby certifies that portions of the above-captioned Petition for *Inter Partes* Review of U.S. Patent 8,934,887, in accordance with and reliance on the word count provided by the word-processing system used to prepare this Petition, that the number of words in this paper is 13,688. Pursuant to 37 C.F.R. §42.24(a), this word count is in compliance and excludes the table of contents, table of authorities, mandatory notices under §42.8, certificate of service, certificate of word count, appendix of exhibits, and any claim listing. This word count was prepared using Microsoft Word.

Respectfully submitted,

/ *Jessica Kaiser* /

Jessica Kaiser
Reg. No. 58,937
Attorney for Petitioner

Date: October 24, 2024
PERKINS COIE LLP
1900 Sixteenth Street, Suite 1400
Denver, CO 80202-5255

CERTIFICATE OF SERVICE

The undersigned hereby certifies that true copies of the Petition for *Inter Partes* Review of U.S. Patent No. 8,934,887 and supporting materials (Exhibits and Power of Attorney) were served via overnight delivery on the Patent Owner at the correspondence address of record as listed on PAIR:

COZEN O'CONNOR
3WTC, 175 Greenwich Street
55th Floor
NEW YORK, NY 10007

A courtesy copy was also sent via electronic mail to Patent Owner's litigation counsel listed below:

Francisco A. Villegas - fvillegas@villegascefo.com
Damir Cefo - dcefo@villegascefo.com
Mark W. Halderman - mhalderman@villegascefo.com
Evan M. Rosenbaum - erosenbaum@villegascefo.com
John V. Picone III - jpicone@spencerfane.com

Respectfully submitted,

/ *Jessica Kaiser* /

Jessica Kaiser
Reg. No. 58,937
Attorney for Petitioner

Date: October 24, 2024