

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Genius Sports Ltd.

Petitioner,

v.

SportsCastr Inc. (d/b/a PANDA Interactive)

Patent Owner.

Case IPR2024-01310

Patent No. 11,039,218

**SUPPLEMENTAL DECLARATION OF AVIEL RUBIN, Ph.D.
IN SUPPORT OF PATENT OWNER'S RESPONSE TO PETITION**

TABLE OF CONTENTS

I.	THE CITED PRIOR ART DOES NOT DISCLOSE OR RENDER OBVIOUS ALL LIMITATIONS OF INDEPENDENT CLAIMS 16 AND 21.....	2
A.	Ground 1 Does Not Disclose Or Render Obvious All Limitations of Independent Claims 16 and 21.....	2
1.	<i>Spivey’s MQSD does not render obvious transmitting information between an “event socket” and a client device.....</i>	2
2.	<i>Regardless of the MQSD’s transmission destination, Petitioner fails to show that Spivey’s topic queues are “sockets.”.....</i>	13
3.	<i>Petitioner combination does not render obvious the claimed “webserver.”.....</i>	17
4.	<i>Petitioner fails to show that a POSITA would have been motivated to combine Ellis with Spivey.</i>	37
B.	Ground 2C: Petitioner Fails to Establish that Herzog in View of Spivey Renders Obvious Every Limitation of the Challenged Claims.....	55
1.	<i>Petitioner’s proposed combination of teachings suffers from many problems.</i>	55
2.	<i>A POSITA would not have been motivated to combine the references’ teachings.</i>	56
II.	CONCLUSION.....	59

EXHIBIT LIST

Exhibit	Description
EX-2001	Declaration of Aviel Rubin, Ph.D.
EX-2002	<i>Curriculum vitae</i> of Aviel Rubin, Ph.D.
EX-2003	Joint Claim Construction Statement, Dkt. 69
EX-2004	Declaration of Dr. Stephen Wicker, September 13, 2024
EX-2005	Yahoo Finance Tanner Simkins, April 20, 2017
EX-2006	NBA.com, Steve Ashburner April 27, 2017 Article
EX-2007	Sportstechie, Robert Roble, April 22, 2017
EX-2008	Sports Tech Tokyo Award; 2019 Yahoo! Sports Tech Awards; Sports Business Awards
EX-2009	Various Media Reports on the Patented Invention
EX-2010	CNBC, December 20, 2018
EX-2011	https://sportscastr.com/about
EX-2012	https://pandainteractive.com/patents
EX-2013	https://sportscastr.com
EX-2014	https://studiopanda.live/
EX-2015	Genius Sports 2-24-18 Email
EX-2016	Genius Sports 5-7-18 Email
EX-2017	'218 Patent Genius Sports Infringement Claim Chart
EX-2018	https://www.opensourceforu.com/2015/03/a-guide-to-using-raw-sockets
EX-2019	https://learn.microsoft.com/en-us/windows/win32/winsock/tcp-ip-raw-sockets-2
EX-2020	Oracle documentation in 2011: Anatomy of the Client/Server Model. https://docs.oracle.com/cd/E13203_01/tux-edo/tux80/atmi/intbas3.htm
EX-2021	Peng, Gang. "CDN: Content distribution network." arXiv pre-print cs/0411069 (2004) https://arxiv.org/pdf/cs/0411069
EX-2022	The History of Streaming Media. https://boyetblog.s3.amazonaws.com/PCPlus/324.Streaming.pdf
EX-2023	RFC 2326: Real Time Streaming Protocol (RTSP) https://www.ietf.org/rfc/rfc2326.txt
EX-2024	Adobe's Real Time Messaging Protocol https://rtmp.veriskope.com/pdf/rtmp_specification_1.0.pdf
EX-2025	WebRTC 1.0: Real time Communication Between Browsers https://www.w3.org/TR/2011/WD-webrtc-20111027/

Exhibit	Description
EX-2026	HTTP Live Streaming: draft-pantos-http-live-streaming-01 https://datatracker.ietf.org/doc/html/draft-pantos-http-live-streaming-01
EX-2027	Microsoft To Add Smooth Streaming in IIS7 https://virtualizationreview.com/articles/2008/11/04/microsoft-to-add-smooth-streaming-in-iis7.aspx
EX-2028	HTTP Dynamic Streaming. (Archived May 14, 2010) https://web.archive.org/web/20100514170442/http://www.adobe.com/products/httpdynamicstreaming/
EX-2029	MPEG ratifies its draft standard for DASH. (Archived December 14, 2011) https://web.archive.org/web/20111214104626/http://mpeg.chiariglione.org/meetings/geneva11-1/geneva_press.htm
EX-2030	Mogulus Rebrands With A Killer Domain: Livestream.com https://techcrunch.com/2009/05/18/mogulus-gets-a-killer-domain-for-rebranding-livestreamcom/
EX-2031	How to watch Obama's inauguration (and the parties) online https://www.cnet.com/tech/tech-industry/how-to-watch-obamas-inauguration-and-the-parties-online/
EX-2032	Twitch is 4 th in Peak US Internet Traffic https://blog.twitch.tv/en/2014/02/05/twitch-is-4th-in-peak-us-internet-traffic-90b1295af358/
EX-2033	Twitch Engineering: An Introduction and Overview https://blog.twitch.tv/en/2015/12/18/twitch-engineering-an-introduction-and-overview-a23917b71a25/
EX-2034	<i>Proceedings of the 25th ACM workshop on network and operating systems support for digital audio and video</i> , pp. 55-60, 2015 https://arxiv.org/pdf/1502.04666
EX-2035	U.S. Patent 11,039,218 Certified File History
EX-2036	Ellis <i>et al.</i> U.S. App. No. 2008/0062318
EX-2037	Shoff <i>et al.</i> US 2001/0001160
EX-2038	DeWeese <i>et al.</i> US 2001/00392.09
EX-2039	Bocharov <i>et al.</i> US 2011/0083144
EX-2040	Institution Decisions for IPR2024-1305, IPR2024-1307, IPR2024-1309, IPR2024-13010, IPR2024-13011
EX-2041	Supplemental Declaration of Aviel Rubin, Ph.D.

Exhibit	Description
EX-2042	Transcript of Deposition of Dr. Stephen Wicker, May 7, 2025

I, Aviel Rubin, Ph.D., declare as follows:

1. I have been engaged on behalf of SportsCastr Inc. (d/b/a PANDA Interactive) (“Patent Owner”), to provide this Supplemental Declaration concerning the technical subject matter relevant to the petition for *Inter Partes* Review 2024-01310 (“Petition”) of U.S. Patent 11,039,218 (“the ’218 patent”).

2. I am over 18 years of age. I have personal knowledge of the facts stated in this Declaration and could testify competently to them if asked to do so.

3. I have reviewed and am familiar with the specification and the claims of the ’218 patent. In general, I will cite to the specification of a United States patent using the following formats: (Exhibit No., Col:Line Number(s)) or (Exhibit No., Paragraph Number(s)). For example, the citation (EX-1009, 1:1–10) points to the ’218 patent specification at column 1, lines 1–10. Also, for convenience, I use italics to denote limitations from the challenged claims.

4. All the opinions in this Supplemental Declaration are based on the documents I reviewed and my knowledge, experience, training and professional judgment. In forming the opinions expressed in this Supplemental Declaration, I reviewed the Petition for *Inter Partes* Review of U.S. Patent No. 11,039,218 and supporting exhibits, Patent Owner’s Preliminary Response and supporting exhibits, as well as the Board’s Institution Decision in this and related IPRs.

5. I am the same Aviel Rubin, Ph.D. who submitted a Declaration as Exhibit 2001 in support of the Patent Owner Preliminary Response submitted in this IPR, which I will refer to as the “POPR Declaration” or EX-2001.

6. For purposes of this Supplemental Declaration, I fully incorporate by reference my entire POPR Declaration, including all qualifications, legal principles, background, state of the art, and opinions presented in that POPR Declaration.

7. In addition to my POPR Declaration, I have been asked to provide the following supplemental opinions.

I. THE CITED PRIOR ART DOES NOT DISCLOSE OR RENDER OBVIOUS ALL LIMITATIONS OF INDEPENDENT CLAIMS 16 AND 21

8. As noted in my POPR Declaration (EX-2001), Petitioner and Dr. Wicker allege that two different combinations—Ellis in view of Spivey and Herzog in view of Spivey—each render obvious independent claims 16 and 21. I disagree with the arguments in the Petition, the opinions set forth by Dr. Wicker, and the Institution Decision for the reasons I discussed in my POPR Declaration, and further for the reasons that I discuss in further detail below.

A. Ground 1 Does Not Disclose Or Render Obvious All Limitations of Independent Claims 16 and 21.

1. *Spivey’s MQSD does not render obvious transmitting information between an “event socket” and a client device.*

9. With reference to claim limitation 16.2 of the ’218 patent, Petitioner’s arguments hinge on its implementation of Spivey’s teachings within Ellis.

Specifically, Petitioner and Dr. Wicker assert that a POSITA would have implemented Spivey's message queue server (comprising the message queue server device and live data server device) in place of Ellis's supplemental content source. (EX-1001, ¶¶106-111.) Dr. Wicker alleges that Spivey discloses a "websocket connection" between the message queue server device's topic queue (the alleged "*event socket*") and client devices, allegedly establishing a "live data connection" (the alleged "*channel*") between the message queue server device's topic queue and the viewer device. (EX-1001, ¶¶105, 109-110, 172.)

10. Spivey only discloses a "websocket connection" as occurring between the client devices and the live data access server 130—i.e., a server which Petitioner and Dr. Wicker have not included in the proposed combination of teachings. (EX-1017, ¶¶43, 61, 74, 92; *see also* EX-2001, Section VI.A.1.) Spivey only discloses the "live data connection" between the client devices and the live data access server, which is not included in Petitioner's proposed combination of teachings. (*Id.*) Spivey does not disclose a connection between the message queue server device and client devices and would not, therefore motivate or teach a POSITA to implement the message queue server device in the way that Petitioner proposes. (EX-2001, Sections VI.A.1.)

11. Indeed, Spivey does not describe the MQSD as capable of connection, e.g., via "websocket connection," with individual client devices. (EX-1017, ¶¶45,

61.) Spivey only describes the LDAS as including the necessary components to support connection of a server with multiple individual client devices using a websocket connection. (EX-1017, ¶¶45, 53-54.) In particular, Spivey describes the LDAS as including, e.g., an “interface component 530.” (EX-1017, ¶59.) The interface component, according to Spivey:

provide[s] user interface elements or supporting data for a client-side user interface. Interface 530 may generally provide an interface to allow an operator to request and authenticate a connection to live data access server 510, to select one or more topics relating to live events for subscription, and to receive and display the live data as it is received by message queue server 120.

(EX-1017, ¶63.) That is, Spivey depicts example “user interfaces” in FIGS. 8-10, which depict, collectively, the process by which a user, via a client device, accesses the data via the LDAS. (EX-1017, ¶¶63, 81-90.)

12. Further, Spivey only describes the LDAS as including, e.g., a “subscription component 540.” (EX-1017, ¶59, FIG. 5.) The subscription component, according to Spivey:

manage[s] subscriptions to topics for currently connected clients. For example, subscription component 540 may receive the selected topic(s) from interface component 530 for a particular connected client. Subscription component 540 may then communicate with message queue server 120 to subscribe to the topic queues that correspond to the selected topics.

(EX-1017, ¶64.) Accordingly, the subscription component facilitates a flow of messages (data) to subscribed client devices. (*Id.*) A client device must first access

the LDAS, and through the LDAS, and specifically, the “subscription component,” subscribed client device may receive data classified according to various “topics.” (*Id.*)

13. Spivey never discusses the MSQ (e.g., MQS 120 of FIG. 1) or the two-server implementation of the MQS (e.g., the MQSD 410 and the LDSD 450 of FIG. 4) as including either the “interface component” or the “subscription component.” (EX-1017, ¶¶ 34-35, 53-58, FIG. 1, 4.) It is my opinion that the interface component and the subscription component are necessary components to facilitate the connection of multiple individual clients to the flow of data described in Spivey. Further, it is my opinion Spivey does not support any of its message queue servers as including the interface component or the subscription component. (EX-2001, Section VI.A.1.) Rather, it is my opinion that the message queue servers, according to Spivey, rely on the LDAS, and the LDAS’s interface component and subscription component to facilitate the message flow described in Spivey. (EX-2001, Section VI.A.1.)

14. Dr. Wicker selectively quotes Spivey to suggest that “client devices ‘request a live data connection’ *with the MQSD [message queue server device]* by ‘select[ing] or enter[ing] the address or connection information *to a particular MQSD.*’” (EX-1001, ¶172 (quoting EX-1017, ¶¶81-82) (emphasis added).) However, Spivey ¶¶81-82 do not support Dr. Wicker’s statement. Dr. Wicker’s

statement is inconsistent with evidence in so far as Spivey ¶¶81-82 describe user interfaces specifically for the purpose of allowing an end client device to access data via the LDAS. Spivey ¶81 could not be more clear:

FIG. 8 illustrates an embodiment of a user interface (UI) 800. UI 800 may be presented by live data app 210. UI 800 may include various UI elements that allow an operator 108 of a client device 110 to **request a live data connection with a live data access server.**

(EX-1017, ¶81 (emphasis added).)

15. Indeed, all of the interfaces shown in Spivey pertain to a client application (e.g., such as that executing on a mobile phone) that accessing a flow of data by requesting a “live data connection” with the LDAS. (EX-1017, ¶81.) Petitioner’s argument rests on the notion that some embodiments of Spivey do not require a LDAS. (Petition, 30-32; EX-1001, ¶¶99, 109-112.) For example, Dr. Wicker states: “one embodiment, the message queue server 120 communicates directly with client devices (110-1...110-a) without the use of a live data access server.” (EX-1001, ¶99.) Dr. Wicker’s position is inconsistent with evidence that Spivey explicitly requires an intervening LDAS to facilitate a connection, such as a websocket connection, between the MQSD and individual client devices. (EX-2001, Section VI.A.1.)

16. Petitioner and Dr. Wicker’s position—that the LDAS is optional—is based on their proposition that the operating environment 400 shown in Spivey FIG.

4 can be divorced from the balance of the teachings of Spivey. (Petition, 30-32; EX-1001, ¶¶99, 109-112.) To the contrary, Spivey explicitly describes the operating environment 400 as a part of the system 100 of FIG. 1. (EX-1017, FIG. 1, 4.) FIG. 4 shows a MQSD and LDAS that may essentially “plug in” as the more generic MQS shown in Spivey FIG. 1. (EX-1017, ¶¶ 34-35, 53-58, FIG. 1, 4.) According to Spivey: “As shown in FIG. 4, message queue server 120 may be implemented with at least two separate devices: a message queue server device 410 and a live data server device 450.” (EX-1017, ¶53.) Thus, as a threshold matter, the operating environment 400 of FIG. 4 should be understood in the context of the overall Spivey system architecture shown in FIG. 1. (EX-1017, ¶¶ 34-35, 53-58, FIG. 1, 4.) Spivey FIG. 1 shows high level system architecture, and FIG. 4 provides addition details pertaining to the MQS shown in FIG. 1. (EX-1017, ¶¶ 34-35, 53-58, FIG. 1, 4.)

17. For purposes of discussion, the operating environment 400 is reproduced below:

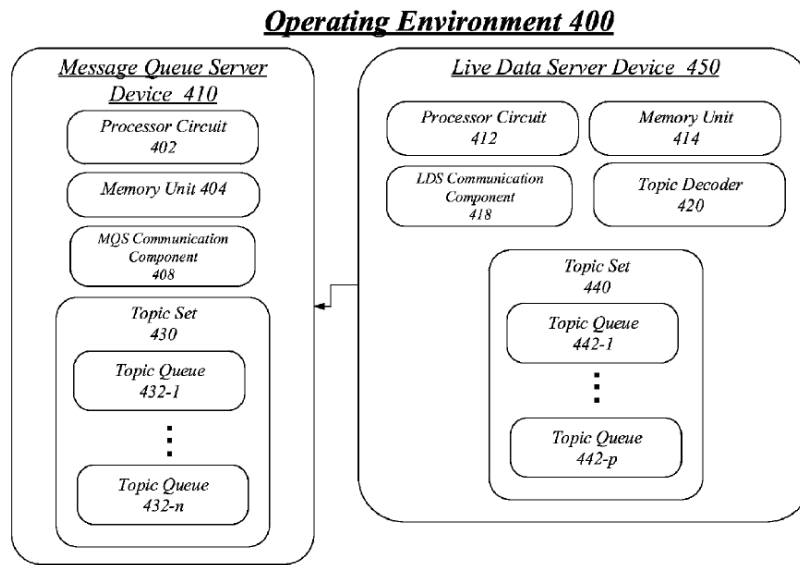


FIG. 4

(EX-1017, FIG. 4.)

18. Absent from FIG. 4 is, e.g., the “interface component” and/or the “subscription component.” (EX-1017, FIG. 4.) Spivey does not disclose any component or functionality of the MQSD and/or the LDSD as being capable of subscribing and connecting to, e.g., via a websocket connection, multiple individual client devices. (EX-1017, ¶¶45, 53, 59-64.) Spivey only describes the LDAS as including this functionality, and the required subscription and interface components. (EX-1017, ¶¶38-39, 59-64.) The MQSD and the LDSD have the ability to communicate with individual end clients, according to Spivey, only via the intervening LDAS. (EX-1017, ¶¶38-39, 59-64.) The MQSD and LDSD may “plug in” as the MQS, and so FIG. 4 is understood only in the context of Spivey high level system 100 architecture, reproduced below:

System 100

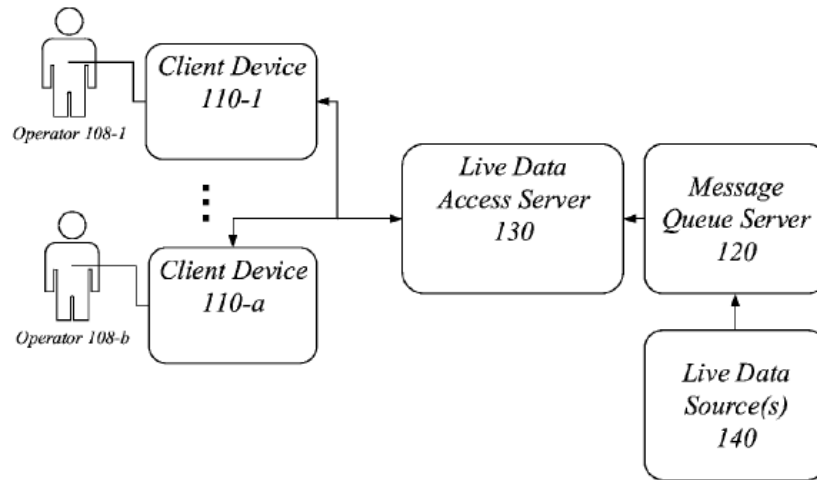


FIG. 1

(EX-1017, FIG. 1.)

19. Accordingly, it is my opinion that Spivey describes the client devices receiving data from the operating environment 400 (e.g., including the MQSD and LDSD) via the LDAS shown in FIG. 1.

20. Not only does Spivey describe the operating 400 only in terms of the system 100 and associated LDAS, but Spivey also never describes the MQSD as having the appropriate functionality to support connections, e.g., via websocket, to multiple individual client devices. (EX-1017, ¶¶45, 53-54, 58.) For example, Spivey never describe the MQSD as a communication endpoint assigned with a publicly-accessible URL or address. (EX-1017, ¶¶45, 53-54, 58.) Nor does Spivey describe the MQSD as a communication endpoint designed to handle real-time

access from a potentially large number of individual clients. (EX-1017, ¶¶45, 53-54, 58.) Nor does Spivey describe the MQSD as a communication endpoint capable of pushing its data packets to multiple individual clients. (EX-1017, ¶¶45, 53-54, 58.) It is my opinion that Spivey does not disclose the MQSD as having the aforementioned functionalities of a communication endpoint because Spivey requires the end client device to “request a live data connection” through the LDAS in order to receive data from the MQSD, which as such, means that Spivey requires the LDAS to enable its message flow.

21. Petitioner and Dr. Wicker’s reliance on Spivey ¶36 does not change my conclusion. Spivey ¶36 states, in relevant part, “[i]n **some embodiments**, message queue server 120 is not accessible to clients over the network directly.” (EX-1017, ¶36 (emphasis added).) However, absent any technical discussion in Spivey of other embodiments, e.g., technical components and associated functionality, Spivey does not suggest other embodiment in which the message queue server is accessible to client over the network directly. (*supra.*) Indeed, the Spivey reference as a whole described the opposite—client devices require the intervening LDAS to access the MQSD—and as such, in my view, the phrase “in some embodiments” would not suggest an alternative technical construct, particularly in view of Spivey’s express teachings to the contrary.

22. As a threshold matter, according to Spivey, client devices are only those “subscribe[d] to and [that] receive live data via live data access server. (EX-1017, ¶39.) Spivey ¶39 contradicts the notion that client devices subscribe to the MQSD.

23. Rather, the only component described in Spivey as permitting communication from the MQSD is the “MQS communication component.” (EX-1017, ¶54.) For example, the MQSD may communication with the MQS communication component 408 (shown in FIG. 4, reproduced below).

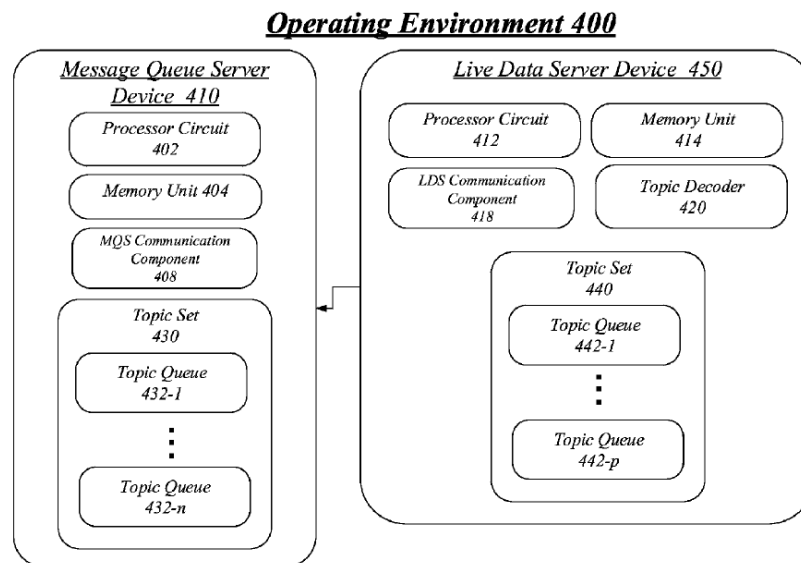


FIG. 4

(EX-1017, FIG. 4.) Spivey describes the MQS communication component 408 as “analogous . . . to . . . MQS communication component 308.” (EX-1017, ¶54.) The MQS communication component 308 (shown in FIG. 3, reproduced below) functions to “push the data payload from the topic queue to a live data access server that has subscribed to the topic of the topic queue.” (EX-1017, ¶49.)

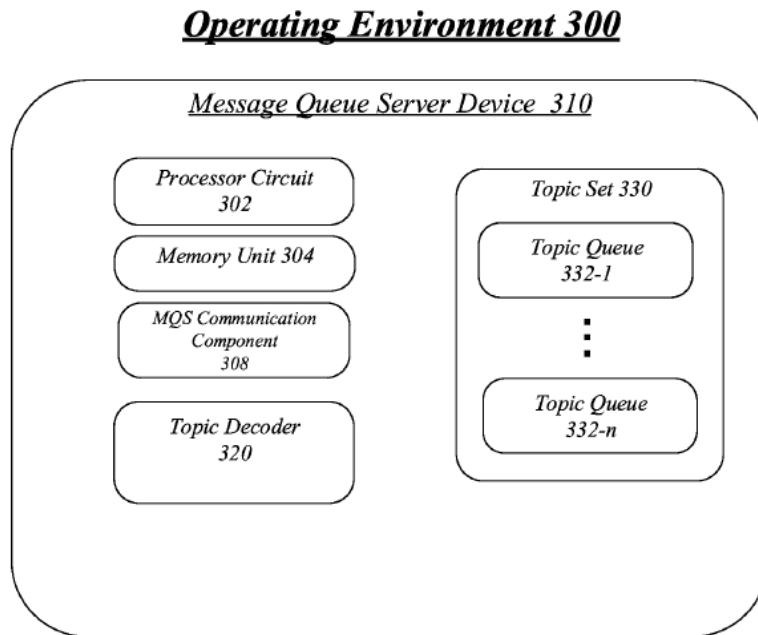


FIG. 3

(EX-1017, FIG. 3). Accordingly, the MQSD—the very server component central to Petitioner and Dr. Wicker’s argument—is only described in Spivey as being capable of communicating with “a live data access server.”

24. The topic flow central to Spivey’s teachings (EX-2001, ¶229) applies to the various message queue servers discussed in Spivey. As I explained in detail herein, the MQSD is one server component of a two-part server environment (the MQSD and LDS of FIG. 4) that essentially “plug in” as the MQS shown in Spivey FIG. 1. (EX-2001, Section VI.A.1.) Spivey discusses a certain “RABBITMQ” as one implementation of the MQSD. (EX-1017, ¶58.) It is my opinion that the RABBITMQ discussed in Spivey ¶58 would function according to the principles set forth in relation to Spivey FIGS. 1 and 4, and indeed, all of Spivey, in so far as the

RABBITMQ would require the LDAS to facilitate connection to numerous individual client devices, according to Spivey. Specifically, Spivey never describes the RABBITMQ as being capable of facilitating a connection, e.g., via a websocket or otherwise, with individual clients, the technical disclosure of Spivey does not support an alternative conclusion. (EX-1017, ¶58.) Rather, Spivey contemplates running the RABBITMQ on the MQSD itself, thereby serving to connect the LDSD and the LDAS. As such, the Spivey disclosure does not support the client devices being connected to the RABBITMQ.

2. ***Regardless of the MQSD’s transmission destination, Petitioner fails to show that Spivey’s topic queues are “sockets.”***

25. Petitioner’s and Dr. Wicker’s argument for equating “topic queues” with “uniquely addressable communication endpoints” rests on Spivey’s FIG. 9. (Petition, 32-34; EX-1001, ¶¶171-172.) However, Spivey’s FIG. 9 depicts a user interface (“UI 900”) as one of a three-part series of user interfaces (e.g., UI 800, 900, 1000 of FIGS. 8-10, respectively) that collectively depicts interfaces for allowing the end user client to receive data from topic queue, expressly via the LDAS and the LDAS’s interface component 530. (EX-1017, ¶81-88.) Dr. Wicker’s reliance on Spivey FIG. 9 for demonstrating uniquely addressable communication endpoints is contrary to evidence that FIG. 9 depicts an interface between the client device and the LDAS. (EX-1017, FIG. 9.)

26. Taking the interfaces in turn, Spivey's FIG. 8 depicts the first user interface in the three-part sequence. (EX-1017, ¶¶81-88.) FIG. 8 particularly shows the visual interface on a client device (e.g., a mobile device) when said client device attempts to log on or otherwise "request a live data connection with a live data access server." (EX-1017, ¶¶81-82.) Spivey describes allowing a client device to request this live data connection using the Live Data Access Server Address 830. (EX-1017, ¶82.) Regardless of whether the Live Data Access Server Address may "resemble a uniform resource locator," the Live Data Access Server Address is used in Spivey only to connect the client device to the LDAS. (EX-1017, ¶82.) The Live Data Access Server Address unequivocally is not used to connect the client device to the MQSD (which sits behind the LDAS), and Spivey does not support such conclusion. (EX-1017, ¶55, FIG. 4.) Accordingly, Petitioner's statement "VCDs 'request a live data connection' with the MQSD" (Petition, 37 (citing EX-1017, ¶¶81-82)) is against the evidence that Spivey ¶¶81-82 only describes and only contemplates connecting the VCSs to the LDAS.

UI 800

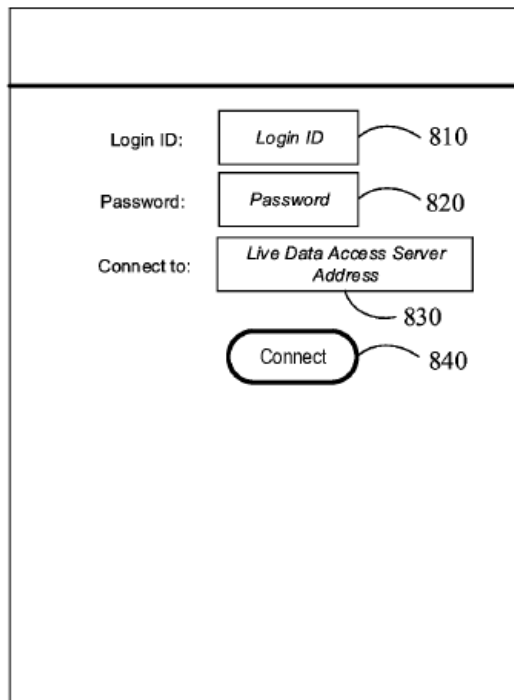


FIG. 8

(EX-1017, FIG. 8.)

27. Turning to the interface of FIG. 9, FIG. 9 particularly shows the visual interface on a client device (e.g., a mobile device) when said client device attempts to indicate to the LDAS that specific data said client device desires to receive. (EX-1017, ¶84, FIG. 9.) The client is permitted to indicate to the LDAS which “topics” it desires to receive by selecting on the UI 900, which topics it desires to receive. (EX-1017, ¶84.) The topics are indicated by text strings, such as “gamecenter.nfl.ravenssteelers,” as indicated in FIG. 9 (reproduced below):

UI 900

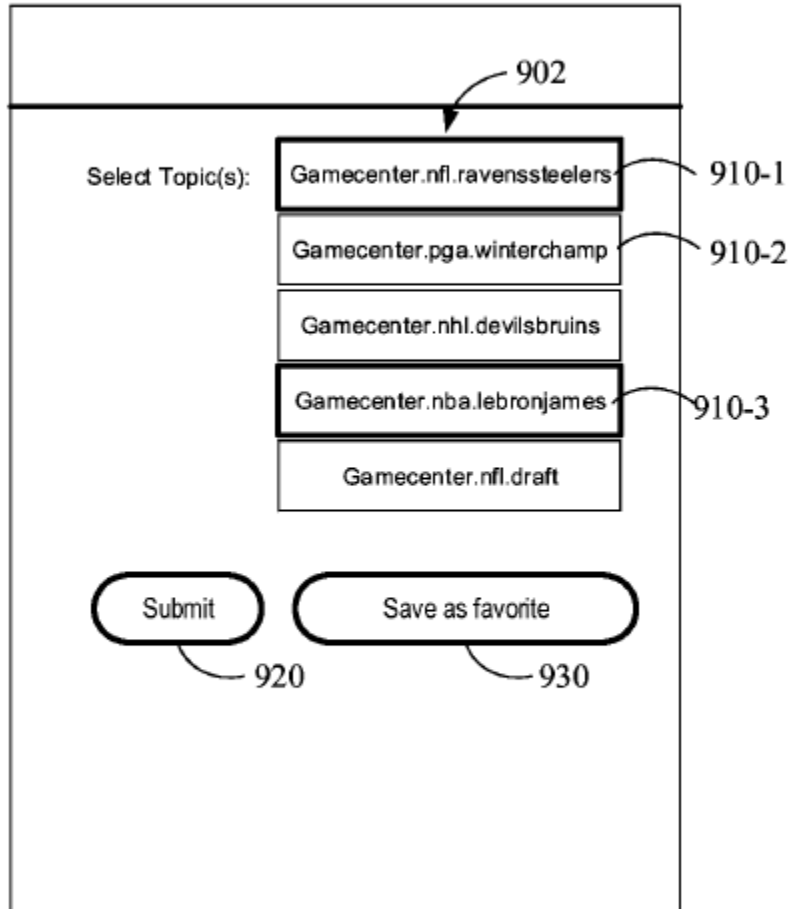


FIG. 9

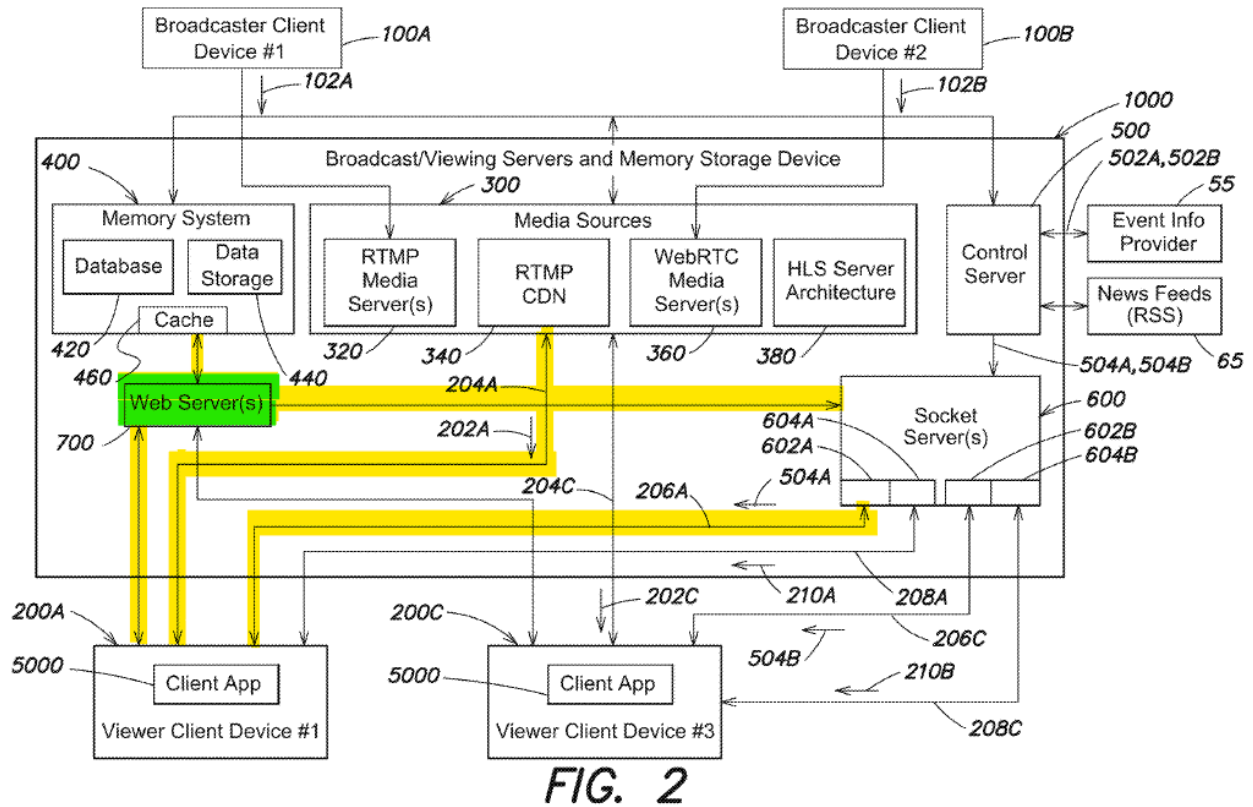
(EX-1017, FIG. 9.) The text strings themselves are not URLs. (EX-1017, ¶84.) Further, text strings are not used in any URL (or any other address) in Spivey. (EX-1017, ¶¶84-86.) A text string or “topic label” is merely that, “a label,” such a label providing an indication of the content of a certain flow of data. A topic label in itself does not suggest a unique connection, nor does it describe a specific technical functionality that would allow a client device to receive said data. A label, is just a

label. Accordingly, without more, Spivey does not suggest that the label shown in FIG. 9 are evidence of uniquely addressable communication endpoints. (EX-1017, ¶¶84-87.) Indeed, Spivey suggest the opposite—that the MQSD is only accessible through the LDAS. (E.g., EX-2001, Section VI.A.1.)

3. ***Petitioner combination does not render obvious the claimed “webserver.”***

28. As shown in FIG. 2 of the '218 patent (reproduced below), a webserver (e.g., webserver 700) may be used to facilitate a connection among numerous client devices, servers, media sources, and/or other components of the system. The configuration of the webserver recited in claim 1 of the '218 patent (and as shown in the example of FIG. 2) may be considered an *orchestrator-type* webserver. Broadly speaking, an orchestrator-type webserver may be used to facilitate connections between various different components of a computer architecture or distributed network. An orchestrator-type webserver, as I describe in greater detail below, is different than a *relay-type* webserver. Broadly, a relay-type webserver may function by serving as an intermediary between different computing modules, itself carrying the data between two modules, whereas an orchestrator webserver facilitates the establishment of a connection between different computing modules so that those computing modules may transmit data between themselves. That is, the orchestrator-type webserver supervises the different computing modules transmit data between themselves such that the orchestrator webserver does not need

to serve as an additional intervening step in the flow of data, which could otherwise slow down the flow of data.



(EX-1009, FIG. 2 (annotated).)

29. In the example of FIG. 2, the webserver 700 may function to facilitate a connection between the viewer client device (e.g., the view client device 200A, 200B) and the media sources 300 and the socket server 600, respectively. In this regard, the webserver 700 supports an architecture whereby the various viewer client devices may connect to, and receive data from, the respective media sources and socket server. As such, the web server 700 itself need not serve as an intermediary or choke point for the flow of data between the respective media sources and the

socket server. The configuration of the webserver 700 therefore shown in the example of FIG. 2 may be considered an orchestrator-type webserver. The orchestrator-type webserver shown in the example of FIG. 2 may be known to have reduced latency as compared with other architectures in which the webserver itself is forced to relay data to other components of the system. For example, it is my opinion that the webserver 700 shown in FIG. 2 (and as recited in limitation 1.3.1) would support the viewer client devices receiving data from the respective media sources and the socket server with less latency because said data need not transport through the webserver itself.

30. The '218 patent describes the webserver with respect to its functionality as an orchestrator-type webserver. For example, “it is via the web server(s) 700 that *client devices create new live streams for broadcast and get access to media servers*, receive access to view other broadcasters' live streams via one of multiple different media sources, *receive event information associated with broadcasters' live streams* and send and receive chat messages, log on and create or update user profiles (or other profiles such as team profiles), and access other social media-related functionality (e.g., digital gift giving) to interact with other users.” (EX-1009, 18:64-19:25.) Accordingly, the webserver facilitates or orchestrates the connection between the client device and the respective media sources (e.g., “client devices create new live streams for broadcast and get access to media servers”) and

socket server (e.g., “receive event information associated with broadcasters' live streams”). In contrast, relay-type webservers, such as those of Spivey/Herzog, as I discuss in greater detail herein, rather than facilitate or orchestrate connections, instead serve as a relay or waypoint for the data enroute to the data’s final destination, which invariably increases processing time due to the presence of additional servers.

31. As another example, the ’218 patent describes the webserver with respect to its functionality as an orchestrator-type webserver in terms of being a “controllable switch”:

As depicted representationally in FIG. 6 , in one aspect the ***web server(s) 700 essentially serve as a controllable switch*** to couple the viewer client device 200A to one of an RTMP media server 320, the RTMP CDN 340 (which is communicatively coupled to the RTMP media server(s) in a virtual private network 330), a WebRTC media server 360 and the HLS serve architecture 360 to receive a copy of broadcaster's live stream of digital content. In the example of FIG. 6 , the ***web server(s) 700 has facilitated a connection between the viewer client device 200A and the RTMP CDN 340.***

(EX-1009, 32:52-61 (emphasis added).) FIG. 6 of the ’218 patent clearly shows that rather than the webserver connect to the media sources, the webserver facilitates the connection of the viewer client device to the media sources, e.g., thereby functioning as a form of an orchestrator-type webserver.

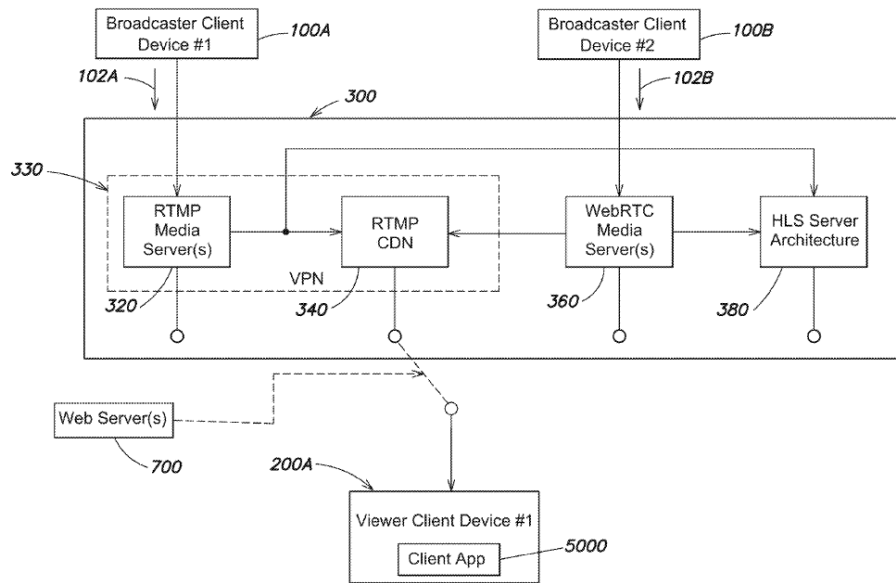


FIG. 6

(EX-1009, FIG. 6.)

32. Limitations 1.3.1-1.3.2 recite features of the webserver described in the '218 patent. For example, limitations 1.3.1-1.3.2 recite a “webserver” (limitation 1.3.1) configured to “transmit, to the first viewer client device” “a first Internet address of a first media source to establish a first video Internet communication channel between the first media source and the first viewer client device to carry the first digital content relating to the first sporting event” (limitation 1.3.1). The recited webserver includes “a first socket address of the first event socket to establish the first event information Internet communication channel to carry the online gaming information” (limitation 1.3.2). Accordingly, a POSITA would understand the recited webserver to function under the rubric of an orchestrator-type webserver.

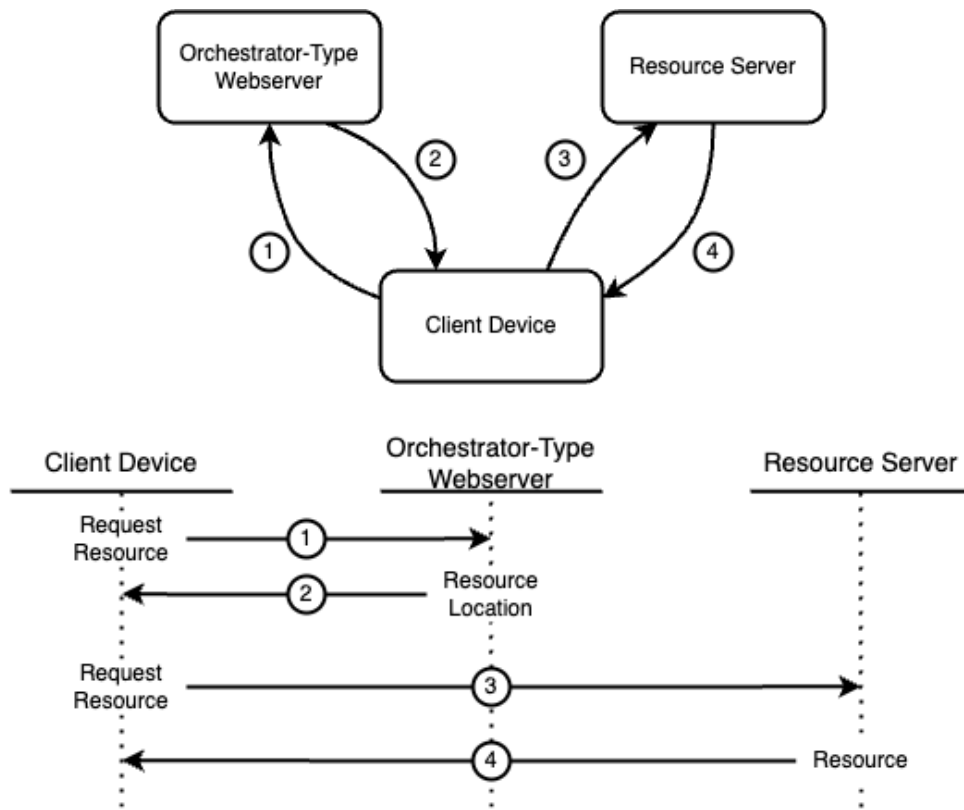
That is, a POSITA would understand the recited webserver to facilitate connections between, e.g., the client devices and the respective media sources and socket server, rather than serve as a delay-inducing intervening stop for data between the client device and respective media source and socket server.

33. With respect to the “webserver,” Petitioner and Dr. Wicket simply argue that a POSA would have found it obvious to integrate a webserver into Ellis/Spivey’s data delivery system and configure it in the claimed manner. (EX-1001, ¶157.) Specifically, Petitioner and Dr. Wicker merely reference their earlier motivation to combine arguments to allege that their Ellis/Spivey combination discloses the recited webserver without articulating specifically how any of alleged servers in Ellis or Spivey would perform the configuration of webserver recited in claim 1, or would otherwise function as an orchestrator-type webserver. (*Id.*)

34. Rather, Petitioner and Dr. Wicker’s argument for Ellis and Spivey teaching the recited webserver is based on a single citation to Ellis ¶40 and Spivey ¶35. (Petition, 34; EX-1001, ¶127.) Petitioner and Dr. Wicker’s assertion that Ellis and/or Spivey disclose an orchestrator-type webserver is against evidence that Ellis and Spivey each disclose a relay-type webserver.

35. As a threshold matter, a POSITA would understand an orchestrator-type webserver to behave in a supervisory role, directing a client device to connect to a secondary resource server. Thus, any data transfers, in this case video content,

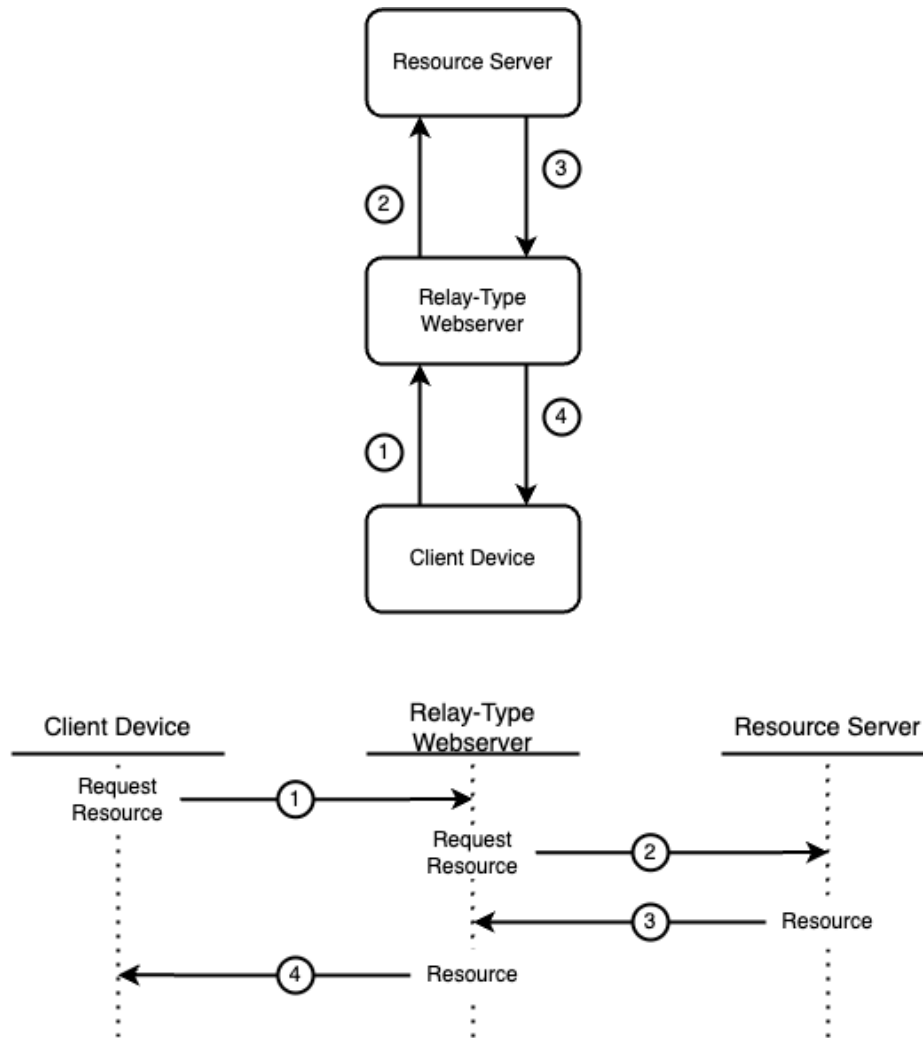
will be transmitted directly from the resource server to the client without having to transit the orchestrator-type webserver. The client will only communicate with the orchestrator-type webserver for control operations, e.g., resource location discovery.



36. The diagram above shows a network diagram depicting a typical orchestrator type webserver and the messages between the various components during a typical resource request. It's clear from this diagram that the orchestrating-type webserver never passes data from the resource server to the client device. Instead, the webserver simply returns the device the location of the resource, e.g., the URL of the resource server. The client device is responsible for retrieving the resource from the resource server directly. In total, the initial connection to the

resource server requires four hops through the network, each hopping inducing a certain amount of latency. However, if the resource requested by the client device is an on-going data stream, e.g., in a streaming video broadcast, all data communications sent set after the initial request will only require two hops through the network, since the data is sent directly from the resource server to the client device. Additionally, since the orchestrator-type webserver is not required for ongoing data transmission between the client and resource server, the connection between the client device and the orchestrator-type webserver can optionally be terminated. Common examples of an orchestrator-type webserver are load balancers, which spreads new incoming clients evenly over a number of different identical servers that are hosting a resource.

37. Further, a POSITA would understand a relay-type webserver to behave as an intermediate between the client device and the resource server. While fulfilling this role, the relay-type webserver will act as the single communication endpoint for both the client device and the resource server. Therefore, the client and the resource server will never directly communication with one another.



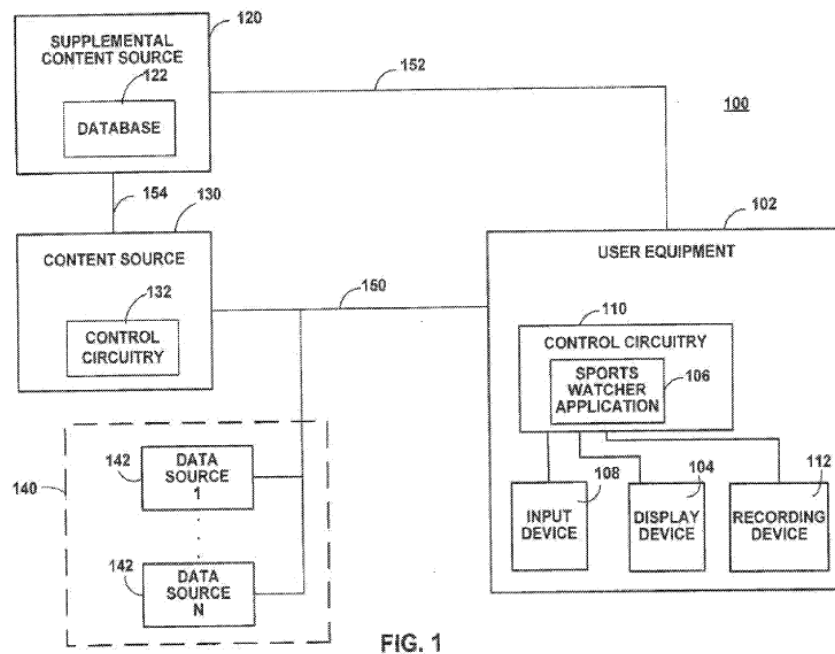
38. The diagram above shows a network diagram depicting a typical relay-type webserver and the messages between the various components during a typical resource request. It is clear from this diagram that the relay-type webserver passes data from the resource server to the client device. The relay-type webserver is induced to make a new, separate request to resource server. This request is on the behalf of the client device, however, the client device is unaware of any connection besides the connection it has with the relaying-type webserver. The resource server returns the requested resource to the relay-type webserver, which in turn returns the

resource to the client device. In total, each data request made by the client device will require four hops through the network, with each hop inducing its own amount of latency. Unlike an orchestrator-type webserver, later data packets will still need to take all four hops to reach the client device. Unlike an orchestrator-type webserver, during ongoing data transmission, e.g., during the serving of a streaming video, the connection between the client device and the relay-type webserver cannot be terminated. Common examples of a relay-type webserver are network edge servers, which provide an entry point into enterprise or service provider core networks.

39. A POSITA would understand that a relay type webserver has several key distinguishing features as compared to an orchestrator type webserver. For example, a relay type webserver, as compared to an orchestrator type will need to handle data messages set from the resource server to the client device and vice versa, will not require the client device to open a secondary connection to the resource server, and will require the a continuous connections between the client device and the relay-type webserver and between the relay-type webserver and the resource server for as long as data is being transmitted between the two.

40. In light of common hallmarks of a relay-type webserver, Petitioner's and Dr. Wicker's position that Ellis ¶40 teaches the recited webserver is against evidence that Ellis concerns a relay-type server functionality. Ellis ¶40 concerns the

content source 130. Ellis's content source does not describe a server component that facilitates connections between disparate server components and the end user. (Section V.A, *supra*.) Indeed, such an orchestrator type connection in Ellis would be counter to the principles of Ellis, which describes a broadcast TV environment in which user equipment "tunes" into content sent from the content source. (EX-1016, ¶75.)



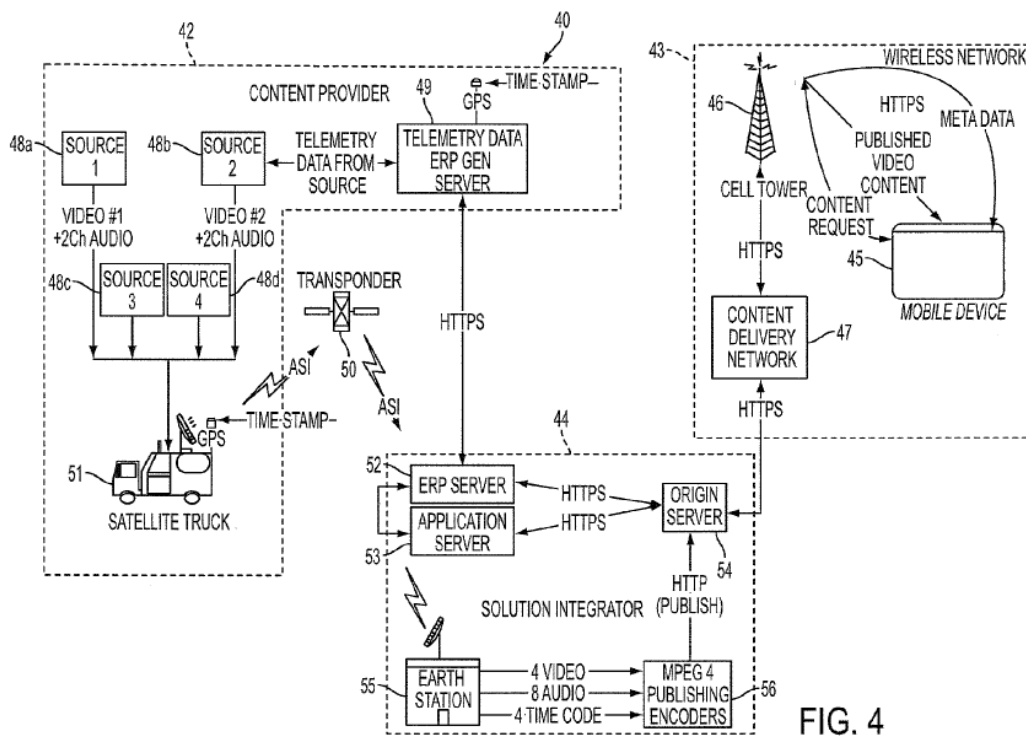
(EX-1016, FIG. 1.) Plainly, the content source does not operate as an orchestrator-type webserver. The content source carries the data, and sends or broadcast the data, to the user equipment. (EX-1016, ¶75, FIG. 1.) The content source does not serve as a facilitator of a connection between the user equipment and other components of the Ellis architecture.

41. In light of common hallmarks of a relay-type webserver, Petitioner's and Dr. Wicker's position that Spivey ¶35 teaches the recited webserver is against evidence that Spivey concerns a relay-type server functionality. Spivey ¶35 concerns the message queue server 120. (*Supra*, EX-1017, ¶35.) As I have described at length herein, and as shown in FIG. 1 of Spivey, the message queue server operates to shuttle or relay data between various different components of the Spivey architecture.

42. Notwithstanding the foregoing deficiencies in the Ellis/Spivey combination, Petitioner's and Dr. Wicker's argues that "to the extent the combination does not disclose a webserver implemented specifically to facilitate communication between a client device and socket server, as well as a client device and media sources, a POSA would have understood that webserver were well-known components in streaming systems that were useful for facilitating communication between client devices and socket servers, and client devices and media sources." (Petition, 34; EX-1001, ¶¶49-53, 127-134.) In support of their proposition, both Petitioner and Dr. Wicker point to Herzog's application server as an illustrative webserver that could apparently facilitate such communication. (*Id.*) However, Herzog's application is also not an orchestrator type webserver either.

43. In light of common hallmarks of a relay-type webserver, Petitioner's and Dr. Wicker's position that the Herzog's application server 53 teaches the recited

webserver is against evidence that Herzog concerns a relay-type server functionality. As shown in Herzog FIG. 4 (reproduced below), the application server is one server along a relay of servers that operate to provide data to the client devices. (EX-1015, ¶48.)

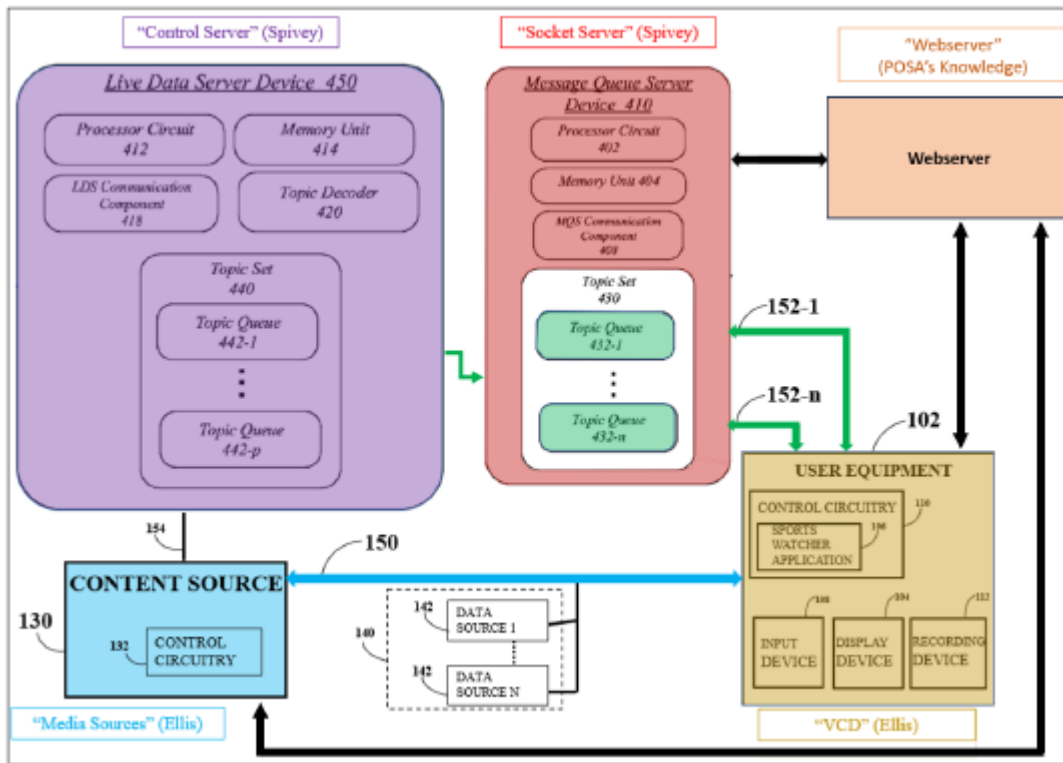


(EX-1015, FIG. 4.) The application server does not operate as an orchestrator-type webserver. Not only does the application server itself act as a relay in that it carries data between, e.g., received telemetry data and the origin, Herzog does not describe in any way using the application server to facilitate a data connection between the mobile device and a source of the data.

44. Fundamentally, Petitioner and Dr. Wicker attempt to put a square peg into a round hole with their reliance on various relay-type webservers of Ellis,

Spivey, and/or Herzog. None of the various alleged webserver pointed to by Petitioner or Dr. Wicker in Ellis/Spivey/Herzog (or indeed anything in Ellis/Spivey/Herzog) is described as being capable of performing the functions of the recited webserver of claim 1 of the '218 patent. Namely, none of Ellis's content source, Spivey's message queue server, and/or Herzog's application server is described as being capable of transmitting instructions to a client device that facilitates a connection between the client device and respective media sources and socket servers, as set forth in limitations 1.3.1-1.3.2.

45. Finally, Petitioner and Dr. Wicker compound their error by attempting to implement an orchestrator-type webserver in its Ellis/Spivey combination, despite the clear teaching of Ellis and Spivey of using a relay-type and not an orchestrator-type webserver. (Petition, 35-36; EX-1001, ¶132-133.)

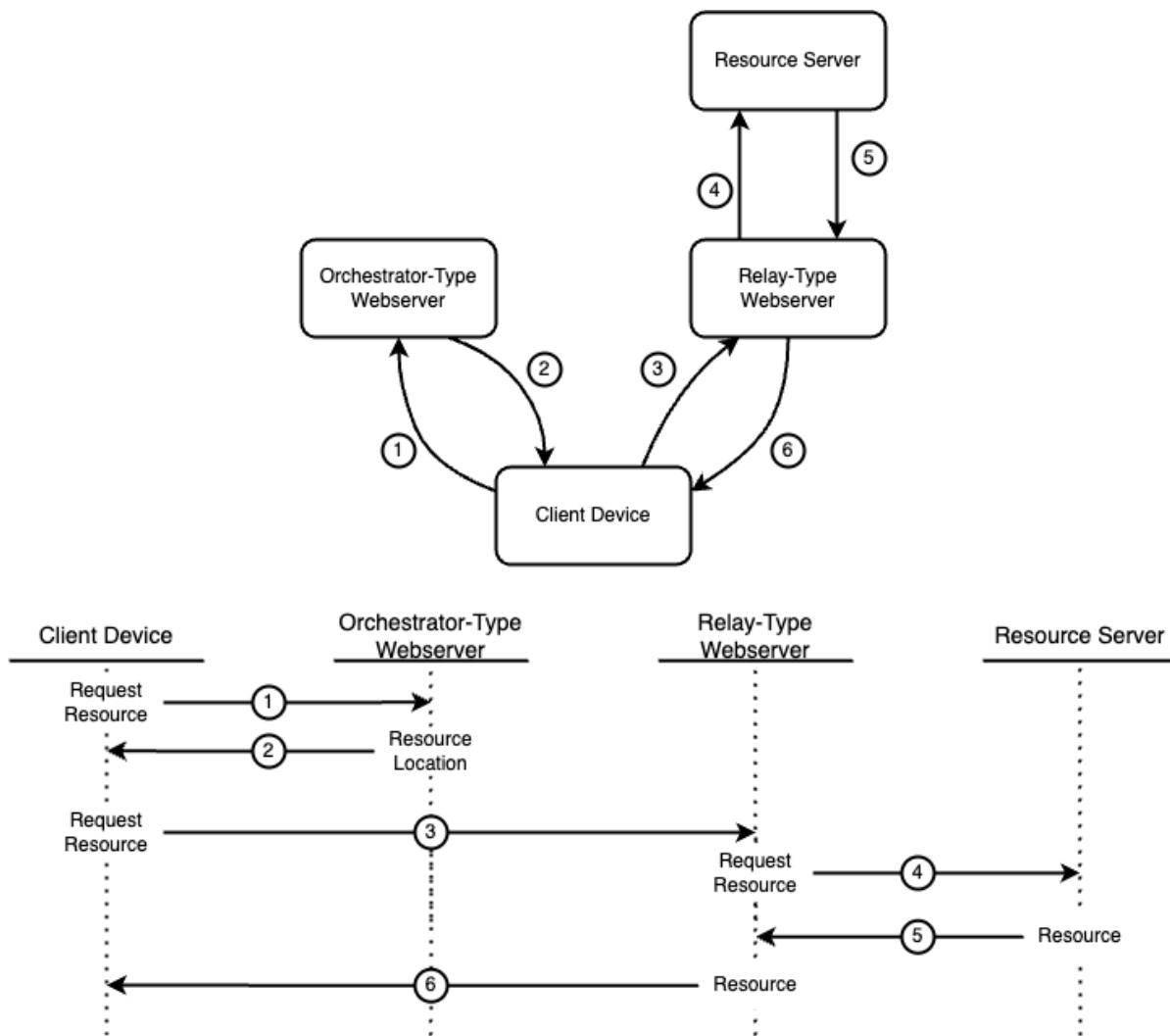


(Petition, 36.)

46. Petitioner's diagram in which it attempts to insert an orchestrator-type webserver into its Ellis/Spivey combination suffers from many problems, which I discuss at length below. (Section VI.A.3, *infra.*) For example, the additional "POSA's Knowledge" webserver would further hinder the operation of Petitioner's Ellis/Spivey combination, leading to additional lag time and complexity because of the presence of additional servers. (Section VI.A.3, *infra.*)

47. In Petitioner's diagram above, the Message Queue Server will be behaving as a relay-type webserver and the "Webserver" Petitioner has fabricated from the POSITA's knowledge would, supposedly, be acting as an orchestrating-type webserver. However, this combination of webserver increases latency in

certain scenarios and adds further complexity and cost to the system while providing not tangible benefit. For instance, if we were to create a general combination of relay-type and orchestrator-type webserver as Petitioner has done here, it would result in the following network diagram and message flow.



48. From this network diagram, we can see that the client device will first have to connect to the orchestrator-type webserver to find the location of the relay-

type webserver, which can then act as a proxy for the client device to reach the resource server (i.e., the Live Data Server Device in Petitioner's figure). Unlike either the orchestrator-type or the relay-type webserver, this combination of servers will require six network hops during a client's initial request for a data resource, during which each hop will add its own specific latency. Thus, this combination of webserver increases the latency of the system in this scenario. During an on-going data stream, e.g., while streaming video, the client will only need to maintain the connection to the relay-type webserver, which will in turn need to maintain a connection to the resource server. Thus, data requests sent during an on-going data stream will require four network hops, the exact same number of network hops, and therefore latency, as Petitioner's diagram without the inclusion of the "Webserver" from a POSITA's knowledge.

49. More fundamentally, however, first Petitioner and Dr. Wicker admit that Ellis/Spivey do not describe, e.g., a webserver "useful for facilitating communication between client devices and socket servers, and client devices and media sources." (Petition, 34; EX-1001, ¶127.) Then, Petitioner and Dr. Wicker fail to establish why a webserver "useful for facilitating communication between client devices and socket servers, and client devices and media sources" would necessarily flow from the teachings of Ellis/Spivey, when these references teach a different genre of webserver entirely, i.e., Ellis/Spivey describe a relay-type

webserver as opposed to an orchestrator-type webserver. It is my opinion that nothing in Ellis or Spivey would suggest that either reference describes the respective end user devices as working concert with an orchestrator-type webserver.

50. For example, with respect to Ellis, Figure 1 appears to indicate direct connections between the User equipment and both the supplemental content source and content source. The text within Ellis only describes these connections simply as “communications paths,” and fails to describe any specific setup process for the creation of these connections. (EX-1015, ¶139.) For this reason, a POSITA would understand these connections as the simplest possible connection, a direct preconfigured connection between the involved devices. A direct connection such as this does not act as an orchestrator-type webserver.

51. As a further example, with respect to Spivey all the components within Spivey are described as being directly connected and relaying messages to the other components that they are connected to. In other words, Spivey only describes relay-type webserver. For instance, Figure 6 which “represent[s] message communicated among the components of system 100” (EX-1017, ¶65) found in Figure 1 is clearly described by Spivey. “Message flow 600 begins when live data is received from live data source(s) 140 at message queue server 120 in message 602 . . . Message queue server 120, upon receiving message 602, may sort the data payload of message 602 into a topic queue, such as topic queue 332-1 or 442-1, according to the topic

label, in message 604. In some embodiments, message queue server 120, e.g. using topic decoder 320, may format the data payload or otherwise alter the structure of the data payload. For example, the data payload may be inserted into an eXtensible Markup Language (XML) structure or into a Javascript Object Notation (JSON) structure.” (EX-1017, ¶¶66-67.) In other words, the MQS receives data directly from the Live Data Source and can potentially alter the information’s form, effectively repackaging the data into a form the Live Data Access Device is expecting it to be in. This is potentially a necessary step since the MQS is acting as a relaying-type webserver, and therefore, the Live Data Access Device is unaware of the any communication the MQS is having with the Live Data Source. If the Live Data Source is providing the video stream in a format different from what the Live Data Access Server is expecting, the MQS must reformat the data before relaying to the Live Data Access Server. "Once the data payload is inserted into a topic queue, message queue server 120 may push the data payload to live data access server 130 in message 606 . . . Upon receipt of message 606, live data access server 130 may push the data payload to client device 110-a in message 608.” (EX-1017, ¶¶68-69.)

52. As a further example, with respect to Herzog, Figure 5 clearly shows a linear network configuration. This means that each component is only able to communicate with the two other components that a directly adjacent it within the network. Consider in this diagram, the path video content takes to reach the client

device as described by Herzog, “a live video content provider 63 feeds multiple live video/audio data stream, by way of a satellite truck 65 . . . A server (for example, a solutions integrator 67) receives the video/audio data as an RF downlink from the satellite truck 65 . . . A wireless provider 69 is configured to receive the video/audio HLS formatted signals and the meta-data HTTPS formatted signals through a content distribution network 68, in which the latter is capable of communicating with the solutions integrator 67 . . . Finally, the mobile device 70 (for example, a video player device, as shown in FIG. 7), which includes a client application program 71, communicates with the content distribution network 68 by way of the wireless provider 69.” (EX-1015, ¶¶59-63.) Rephrased more directly, the live video content provider 63 will pass the feed to a satellite truck 65, which will relay the data to the solution integrator 67, which will relay the data to the content distribution network 68, which will relay the data to the wireless provider 69, which will relay the data to the mobile device 70 containing the client application 71. Nowhere in Herzog does a component facilitate the direct connection between two different components. Each component, as clearly demonstrated in Figure 5, relays information between each of its neighbors within the network.

53. Accordingly, when faced with Petitioner’s Ellis/Spivey combination, a POSITA would not find either reference to include, nor find it necessary to add, an orchestrator-type webserver to Petitioner’s proposed Ellis/Spivey architecture.

Ellis/Spivey/Herzog each function without an orchestrator type webserver in concert with the end device, and so, adding such webserver to Petitioner's diagram at best would be superfluous and unnecessary. And as I describe in greater detail below, the addition of such webserver would likely hinder the performance of the proposed combination.

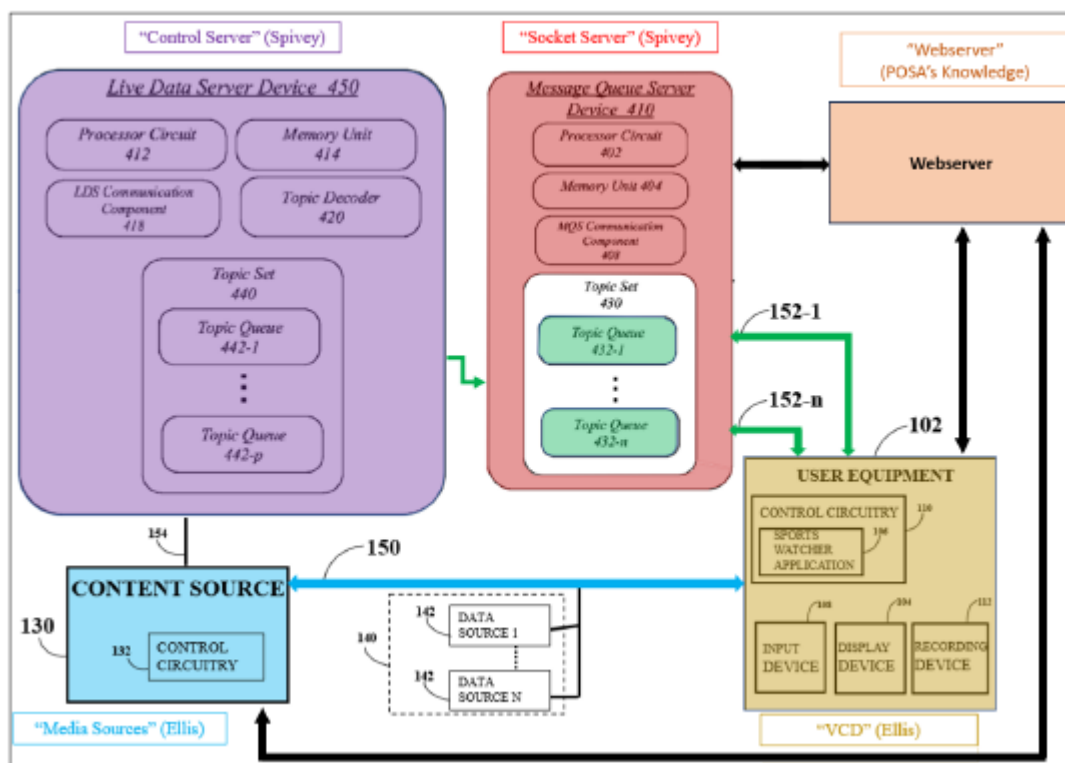
54. Consequently, the Ellis/Spivey/Herzog documents do not disclose a webserver configured to perform limitations recited in claim 1 (e.g., to facilitate or orchestrate connections with the client devices and other components), and Petitioner's proposed addition of such orchestrator-type webserver to its Ellis/Spivey diagram would not be obviously attempted for at least because the addition of a different genera of server (a webserver) to the Ellis/Spivey relay-type framework would be superfluous and unnecessary to the operation described in Ellis and/or Spivey.

4. ***Petitioner fails to show that a POSITA would have been motivated to combine Ellis with Spivey.***
 - a. **Petitioner's proposed combination radically modifies Spivey's teachings.**

55. Further, Petitioner and Dr. Wicker implicitly acknowledge that the Ellis/Spivey combination, which I discussed above (*supra*), may not disclose "a webserver implemented specifically to facilitate communication between a client device and socket server, as well as a client device and media sources." (Petition, 34; EX-1001, ¶127.) Instead, Petitioner relies on the opinion of Dr. Wicker insofar

as Dr. Wicker states: “in my opinion, a POSA would have utilized a known webserver to facilitate such communication.” (*Id.*)

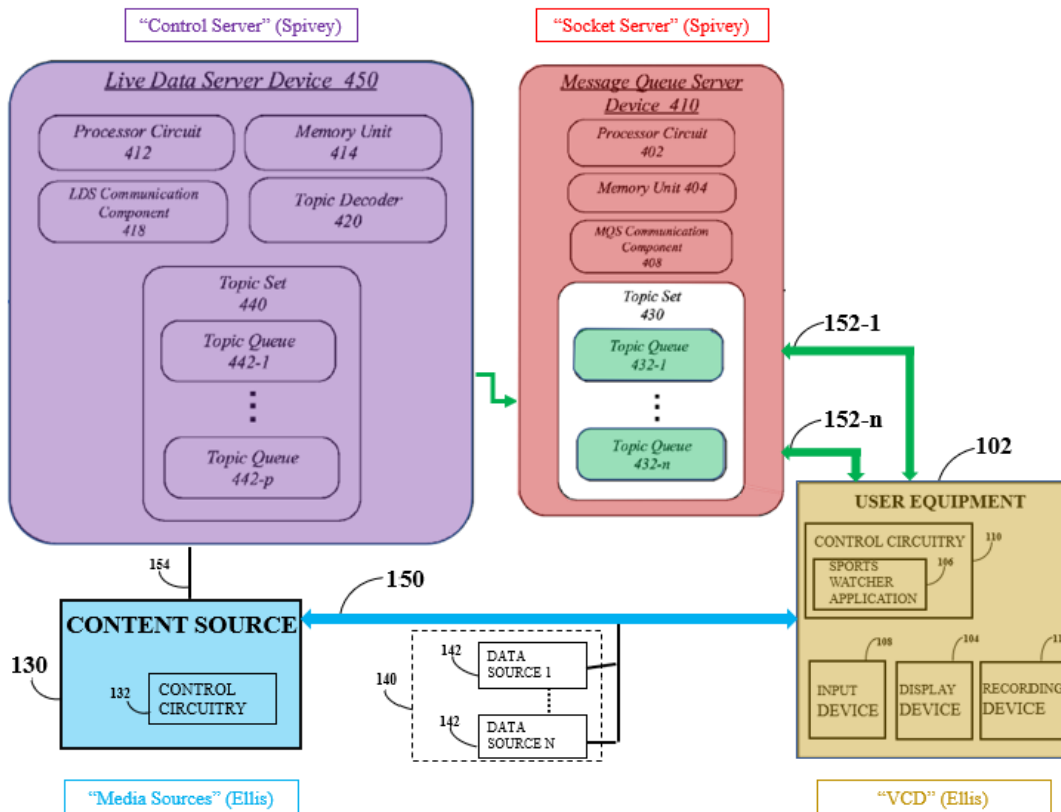
56. Dr. Wicker attempts to support his position by stating that “webservers were well-known components in streaming systems that were useful for facilitating communication between client device and socket servers, and client device and media sources.” (EX-1001, ¶128.) Dr. Wicker also points to Herzog’s application server as an illustrative webserver, as I discussed at length herein. (EX-1001, ¶132; *supra.*) Dr. Wicker then purports to make up for the deficiency of the Petitioner’s Ellis/Spivey combination by further annotating their hypothetical diagram with a webserver of Petitioner’s and Dr. Wicker’s own making:



(Petition, 36.)

57. Again, rather than apply the teachings of Ellis and Spivey as they are, the proposed combination attempts to add a superfluous *orchestrator-type* webserver to a system that relies on a different type of webserver, a *relay-type* webserver, as taught in Ellis and Spivey. Neither Petitioner or Dr. Wicker explain how an orchestrator-type webserver would be relevant to or even work in the context of Ellis's and Spivey's relay-type webserver teachings. Nor do Petitioner or Dr. Wicker explain the appropriateness of selecting an orchestrator-type webserver configuration (from all possible computer architecture at the command of the POSITA) for insertion into an incompatible system.

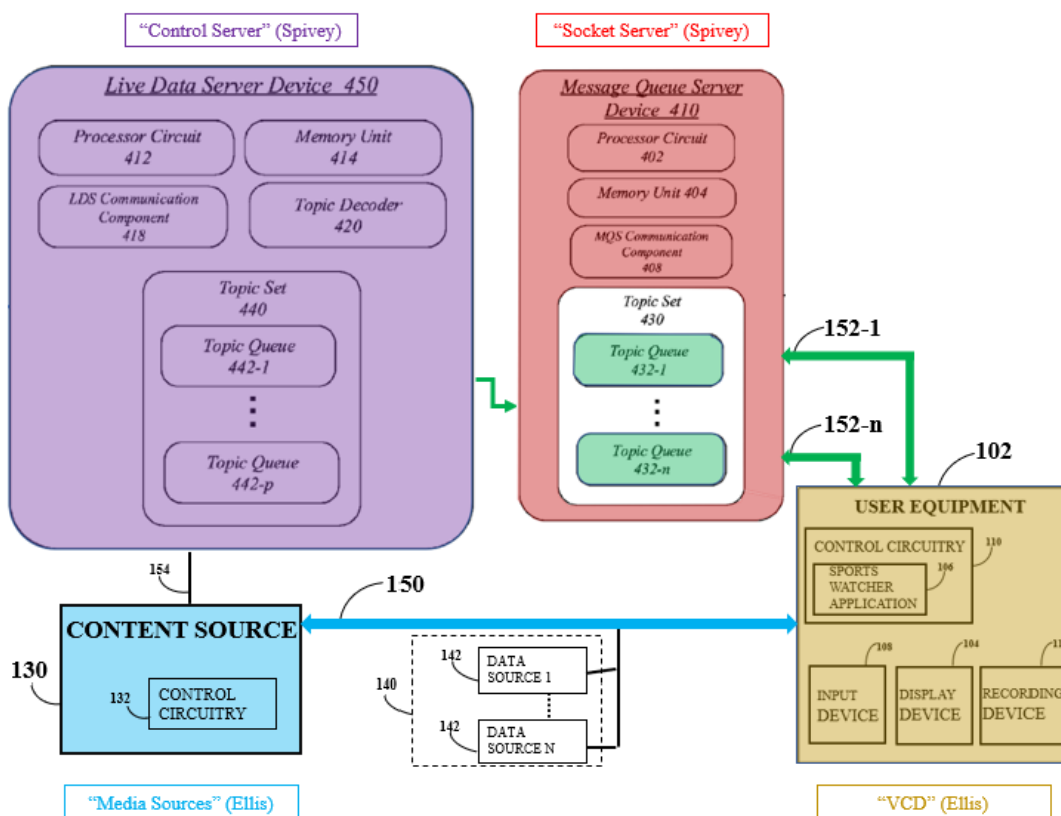
58. As I described at length above, Ellis does not describe use of an orchestrator-type webserver. (Section VI.A.1, *supra*.) Rather than use such orchestrator-type webserver, Ellis broadcasts data to an end user who “tunes” into the broadcast content. (Section V.B, *supra*.) Such relay-type server principles are illustrated in Petitioner's original Ellis/Spivey diagram (reproduced below), in which a “content source” relays data to “user equipment” without an additional orchestrator-type webserver. Petitioner shows this data relay with use of blue annotated line 150.



(EX-1001, ¶122.) Accordingly, regardless of Dr. Wicker’s statement that webserver are “well-known,” Ellis does not implement the specific type of webserver Petitioner proposes in its updated diagram. That is, Ellis does not describe an orchestrator-type webserver, and so Petitioner’s addition of such orchestrator-type webserver has no basis in Ellis’s teachings. (Section VI.A.1, *supra*.)

59. Also, I described at length above, Spivey does not describe use of an orchestrator-type webserver. (Section VI.A.1, *supra*.) Rather than use such orchestrator-type webserver, Spivey shuttles data along a relay of servers—from the MQS, to the LDAS, to the client devices—and thereby avoids a direct connection

between the client devices and any data source. Such relay-type server principles are illustrated in Petitioner’s original Ellis/Diagram (reproduced below), in which a “Live Data Server Device” relays data to a “message queue server device” without an additional orchestrator-type webserver. Moreover, while Petitioner’s diagram is incorrect, as I explained at length above, in that the Spivey message queue server device is not described as connecting to client devices, Petitioner nevertheless still shows the message queue server device as relaying data (without a facilitating webserver) to client device.



(EX-1001, ¶122.) Accordingly, regardless of Dr. Wicker’s statement that webserver are “well-known,” Spivey does not implement the specific type of

webserver Petitioner proposes in its updated diagram. That is, Spivey does not describe an orchestrator-type webserver, and so Petitioner's addition of such orchestrator-type webserver has no basis in Spivey's teachings.

60. Dr. Wicker's argument for incorporating an additional webserver into Petitioner's Ellis/Spivey combination is based on Spivey's use of the word "web server" in a long list of example electronic devices. However, little technical information can be gleaned from Spivey ¶35:

Examples of ***an electronic device may include***, without limitation, an ultra-mobile device, a mobile device, a personal digital assistant (PDA), a mobile computing device, a smart phone, a telephone, a digital telephone, a cellular telephone, ebook readers, a handset, a one-way pager, a two-way pager, a messaging device, a computer, a personal computer (PC), a desktop computer, a laptop computer, a notebook computer, a netbook computer, a handheld computer, a tablet computer, a server, a server array or server farm, ***a web server***, a network server, an Internet server, a work station, a mini-computer, a main frame computer, a supercomputer, a network appliance, a web appliance, a distributed computing system, multiprocessor systems, processor-based systems, consumer electronics, programmable consumer electronics, game devices, television, digital television, set top box, wireless access point, base station, subscriber station, mobile subscriber center, radio network controller, router, hub, gateway, bridge, switch, machine, or combination thereof.

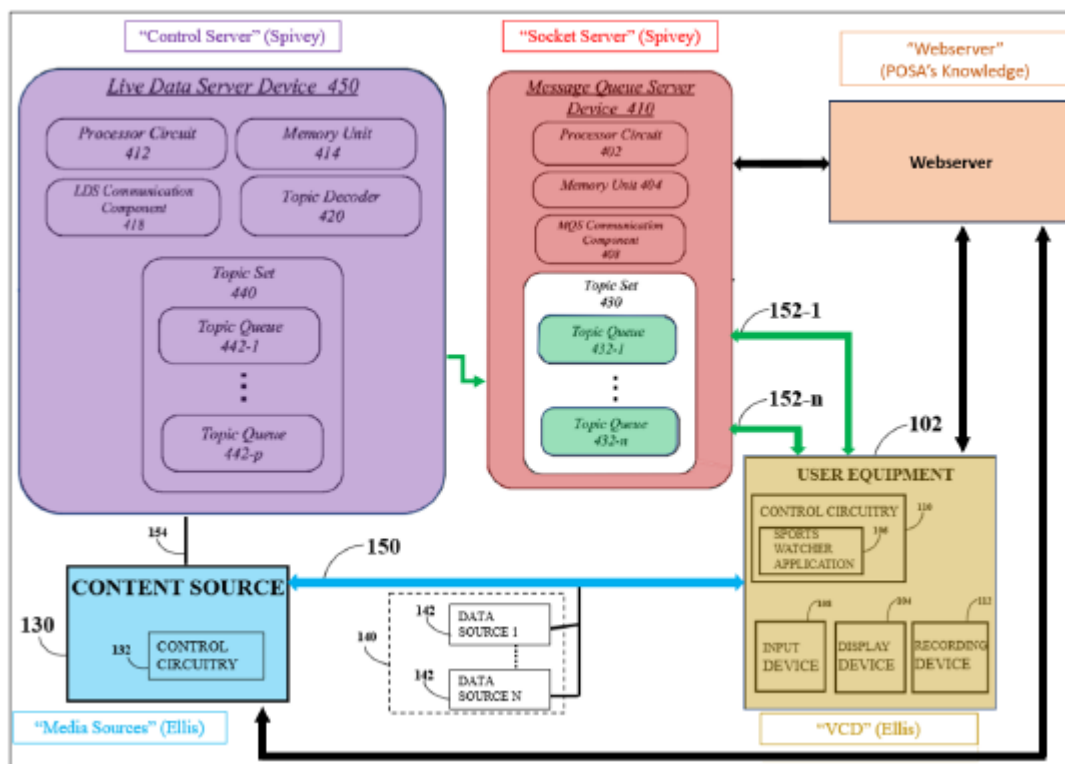
(EX-1017, ¶35 (annotated).) The mere fact that Spivey uses the word "web server" as one example in long list of generic devices does not support that Spivey contemplates integrating its techniques with an orchestrator-type webserver. Indeed,

Dr. Wicker's assertion that Spivey could be implemented with such orchestrator-type webserver is against evidence that Spivey relies exclusively on its disclosure relay-type webserver techniques. (Section VI.A.1, *supra.*) Accordingly, to the extent that Spivey ¶35 can be used as applicable whatsoever, a POSITA would understand Spivey ¶35 to refer to a relay-type webserver, consistent with Spivey's teachings of relay data from one server to the next. (*Id.*)

61. As I have discussed herein, the relay-type webserver techniques found in Ellis and Spivey differ from orchestrator-type webserver techniques found in the '218 patent. (Section VI.A.1, *supra.*) Ellis and Spivey each function by including server components that relay or shuttle data from one component to another. (*Id.*) Absent from either Ellis or Spivey is an orchestrator-type webserver that facilitates a data connection, rather than itself serve as a waypoint for the data. (*Id.*) Petitioner and Dr. Wicker are proposing to modify the Ellis/Spivey teachings of a relay-type webserver with an orchestrator-type webserver, not found in either reference. As such, Petitioner and Dr. Wicker are attempting to change a core operation of each of Ellis and Spivey and create a whole new computer architecture not found in either reference.

62. For example, not only do Petitioner and Dr. Wicker propose to modify the relay-type webserver techniques of Ellis/Spivey, but Petitioner and Dr. Wicker go even further and propose to add an orchestrator-type webserver to somehow

supplement the relay-type server techniques of Ellis and Spivey. There is no basis in Ellis or Spivey of supplementing relay-type architectures with an additional, superfluous webserver. I again reproduce Petitioner's diagram, which inappropriately and unexplainedly includes both relay-type and orchestrator-type architectures:



(Petition, 36.) As is shown in this diagram, Petitioner proposes to maintain the relay-type techniques of Ellis and Spivey and additionally include an orchestrator-type webserver. Petitioner and Dr. Wicker completely fail to explain the purpose of the additional orchestrator-type webserver. Petitioner and Dr. Wicker further fail to explain the interaction of the additional orchestrator-type webserver with the data-relay techniques of select Ellis/Spivey components.

63. In so doing, Petitioner and Dr. Wicker have created a diagram that implements a superfluous server with no apparent purpose. As such, Petitioner and Dr. Wicker's diagram creates a system that suffers from many problems, including latency (due the presence of additional servers), as I explain in greater detail herein.

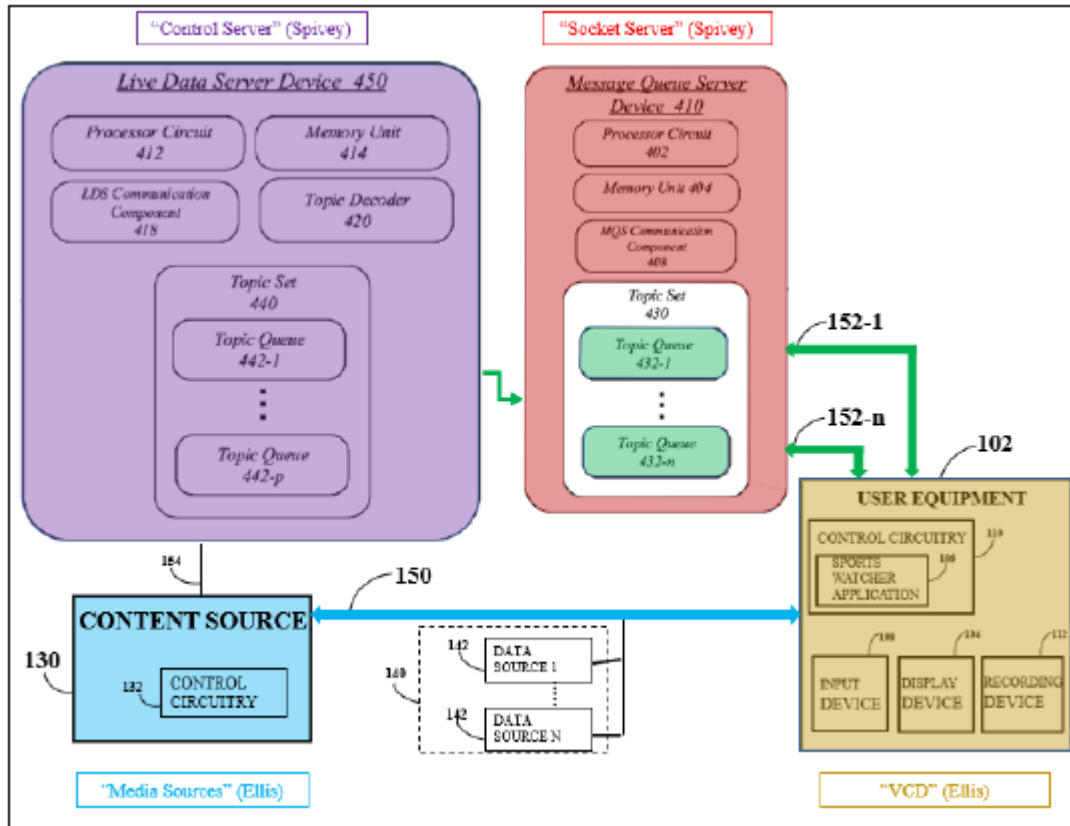
b. Petitioner's proposed combination of teachings suffers from many problems.

64. With initial reference to the webserver, even assuming one would select an orchestrator-type webserver as an addition to the Ellis/Spivey teaching, the combination proposed by Petitioner and Dr. Wicker suffers from many problems.

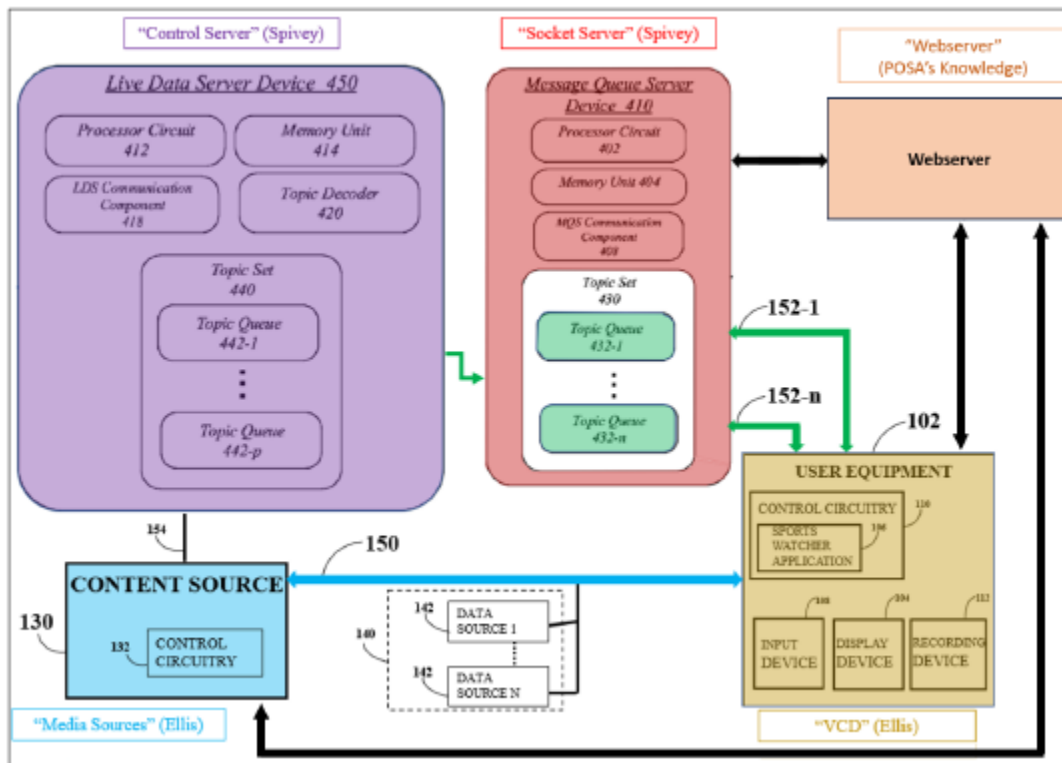
65. *For example*, neither Petitioner nor Dr. Wicker have attempted to explain how the proposed combination would operate with **both** the relay-type server techniques of Ellis/Spivey and the orchestrator-type webserver drawn in the diagram of Petitioner's own making. Fundamentally, each server component added to a hypothetical computer diagram may cause some additional delay. For example, each server component may perform many actions with data, including receiving, processing, and/or transmitting data, each of which may cause a certain delay. A POSITA would therefore understand that regardless of the purpose of Petitioner's inclusion of the additional orchestrator-type webserver, Petitioner's insert of such would be likely to cause delays.

66. For example, modifying Petitioner's initial diagram (Petition, 22 (reproduced below) by adding an additional webserver (Petition, 28 (reproduced

below) would be understood by a POSITA to increase delay to the presence of an additional server component.



(Petition, 22.)



(Petition, 36.)

67. To the extent that the addition of the “webservice” would cause delay, neither Petitioner nor Dr. Wicker describe why such delay would be desirable.

68. Moreover, the additional “webservice” is simply unnecessary to the Ellis/Spivey combination. As I describe herein, Ellis and Spivey each describe transmitting data under a relay-type framework. (Section VI.A.1, *supra*.) As a general principle, a relay-type webservice and an orchestrator-type webservice operate independent of one another. That is, a relay-type webservice does not require an orchestrator-type webservice to operate and vice versa. Accordingly, a POSITA would not understand the addition of Petitioner’s orchestrator type webservice as necessary to the operation of either Ellis or Spivey. Quite the opposite, a POSITA

would understand the addition of such orchestrator-type webserver as redundant and server no purpose that would necessarily support the relay-type operations described in Ellis and Spivey.

69. *As another example*, Petitioner’s and Dr. Wicker’s argument that a POSITA would have had “a reasonable expectation of success in integrating a webserver into Ellis and Spivey’s data delivery system” is inconsistent with evidence that Ellis and Spivey disclose a completely different, relay-type server architecture as compared to Petitioner’s created diagram. Petitioner and Dr. Wicker do not propose some generic version of a “webserver.” (Petition, 34-37; EX-1001, ¶¶127-134.) Rather, Petitioner and Dr. Wicker propose a very specific computer architecture in which they illustrate a webserver as an orchestrator. (*Id.*) A POSITA would not have a reasonable expectation of success because a POSITA would not understand how an orchestrator-type webserver would be applicable to the relay-type server techniques of Ellis and/or Spivey. Moreover, a POSITA would understand that at minimum the addition of another server component—the proposed “webserver”—would be an additional component for interacting with data in some fashion, and thus likely to add to system latency as previously described. Neither Dr. Wicker nor Petitioner describe why additional latency would be desirable (as may occur with the presence of additional server components). Accordingly, understanding the proposed combination to hinder system

performance (e.g., due to redundancy and increased latency), a POSITA would not be expected to succeed in creating such combination.

70. Turning next to Spivey's message queue server, even assuming one would use Spivey's other teachings, the combination proposed by Petitioner and Dr. Wicker suffers from many additional problems.

71. Notably, Petitioner's diagram (Petition, 36) does not include any component to support the relay of data between the MQSD (of Spivey) and the user equipment (of Ellis). Spivey never describes the MQSD operating in isolation or apart from the LDAS. (EX-2001, Section VI.A.2.) Indeed, the MQSD shown in Petitioner's diagram does not include any component described in Spivey as being capable of transmitting information to multiple individual client devices. (EX-2001, Section VI.A.2.) The only component shown in Petitioner's diagram that is capable of transmitting information whatsoever from the MQSD, is the "MQS Communication Component 408." (*Compare* Petition, 36 with EX-1017, ¶54, FIG. 4.) However, Spivey only describes the MQS Communication component as being capable of transmitting information to a LDAS, which is an intervening server between the MQSD and any individual client devices. (EX-2001, Section VI.A.2.) Accordingly, Petitioner's diagram suffers from a fundamental problem insofar as the diagram does not provide any support for a transmission mechanism of data from the MQSD to the user equipment.

72. Petitioner and Dr. Wicker attempt to sidestep the apparent inoperability of the Petitioner's diagram, by illustrating multiple "communication channels" 152-1, 152-n in this diagram. (Petition, 36; EX-1001, ¶52.) However the notion of multiple "communication channels" stemming from the MQSD is a concept entirely of Petitioner's and Dr. Wicker's own making. For example, Ellis never describes multiple paths. (*See generally* EX-1016.) And Spivey never describes multiple connections from the MQSD. (EX-1017, ¶¶45, 53-58, FIG. 4.) Petitioner's inclusion of "152-1, 152-n" attempts to suggest that the MQSD is capable of handling communication to multiple individual clients. However, Spivey does not describe the MQSD as having this functionality. (EX-2001, Section VI.A.2.) Spivey MQSD requires the intervening LDAS in order to send data to individual clients. (*Id.*) Accordingly, despite Petitioner drawing additional communication channels in its diagram, the MQSD that Petitioner adopts from Spivey's disclosure would not support these multiple communication channels. In this regard, Petitioner's combination would not be operable to transmit data to user equipment not only because the MQSD requires the intervening LDAS, but also because the MQSD is not equipped itself to handle communication to multiple individual clients, as I have explained in detailed herein. (*Id.*)

c. A POSITA would not have been motivated to combine the references' teachings.

73. Petitioner relies on a belief that “a POSA would have appreciated that Spivey’s robust framework could easily be integrated with Herzog’s webserver to facilitate client device communication because Spivey broadly claims that nearly any electronic device can serve that purpose.” (Petition, 37.) However, Spivey merely lists the word “web server” as one illustrative electronic device in a list of dozens of electronic devices. (EX-1017, ¶35.) A generic listing of a webserver does not support a finding that Spivey teaches or suggest the specific webserver proposed by Petitioner and Dr. Wicker, which is that of an orchestrator-type webserver. (Petition, 37; EX-1001, ¶¶116-119.) Indeed, the generic listing of a “web server” in Spivey must be understood within the context of the entire Spivey document to be a relay-type webserver—this is Spivey’s main teaching, relaying data from the message queue server device, to the live data access server, to the client devices. Accordingly, a POSITA would have not appreciated that Spivey’s framework would be operable, compatible or modifiable with a different genre of webserver.

74. Moreover, a POSITA would not be motivated to add an additional webserver (an orchestrator-type or otherwise) to the Ellis/Spivey Petitioner proposal because a POSITA would understand that doing so would add additional components, and therefore add complexity, cost, and create latency. For example, and as I have explained herein, each additional server component may create latency

to the extent that server component may be used to receive, process and/or handle data. Therefore, absent some articulated rationale as to why the additional server component would improve performance, a POSITA would assume that the additional server component indeed would detract from the performance by increasing system latency. Seeing the increased latency as undesirable, a POSITA would therefore not be motivated to make the proposed combination of the additional, orchestrator-type server into the Ellis/Spivey system.

75. Yet further, a POSITA would not be motivated to add an additional, orchestrator-type webserver because doing so would serve no apparent purpose, in light of teachings of Ellis and Spivey. Ellis and Spivey each operate using a relay-type approach to handling data. A relay-type approach to handling data does not require an additional, orchestrator-type webserver to operate. And indeed, neither Ellis nor Spivey describe their relay-type techniques as requiring one either. Accordingly, a POSITA would not only find the addition of another webserver as prone to increase latency, but also a POSITA would not find the addition of another webserver as serving any purpose whatsoever, particularly in light of the relay-type teachings of Ellis and Spivey. Seeing the addition of another webserver as serving no apparent purpose, a POSITA would find such webserver redundant, and would not be motivated to make such combination.

76. Additionally, Dr. Wicker’s primary reason that a POSITA would have allegedly implemented Spivey’s teachings within Ellis’s system is to minimize latency. (EX-1001, ¶¶108-109.) Specifically, Dr. Wicker asserts a POSITA “looking to improve Ellis’s latency issues—would have been drawn to a system such as Spivey” in order to “achieve full-duplex communications between a client and a remote host while minimizing latency.” (EX-1001, ¶109.) There are many errors in this reasoning.

77. Ellis does not disclose any issues with latency or delay, and neither Petitioner nor Dr. Wicker explain why Ellis’s system would have been understood to suffer from such issues or the cause of such issues. Dr. Wicker refers to “Ellis’s latency issues” without actually explaining why or how a POSITA would have understood Ellis to suffer from latency issues. (EX-1001, ¶109.) While the Ellis reference claims priority on its face back to 2006, the application that resulted in the Ellis reference was filed in 2014, several months before the application for Spivey was filed. (EX-1016, p.1; EX-1017, p.1.) Dr. Wicker claims that applying the limited, modified teachings of Spivey would achieve transmission to client devices in a period of time that “is at least two orders of magnitude shorter than conventional time periods of legacy devices.” (EX-1001, ¶110 (quoting EX-1017, ¶96).) But Ellis is not a legacy device, nor has Dr. Wicker shown it to be.

78. Furthermore, all of Spivey's disclosures of a "full-duplex connection, such as a websocket connection" that purportedly minimized delay, describe this type of connection as occurring between the live data access server 130 and the client devices. (EX-1017, ¶¶43, 61, 74, 92.) But Petitioner's proposed combination of teachings ignores the live data access server, instead incorporating only the message queue server. While this message queue server includes a live data server device, Spivey describes this device as being on the non-public, non-client side (EX-1017, ¶57), which is also how Petitioner has proposed use of the 2-component message queue server (Petition, 19-24). That is, applying Spivey's limited, modified teaching in the way that Petitioner suggests does not include the "full-duplex connection" Spivey discloses. Accordingly, a POSITA would not understand Petitioner's Ellis/Spivey combination to achieve a latency reduction, e.g., of "two orders of magnitude" without implementing Spivey teachings, as a whole, which expressly include the LDAS. Petitioner does not include the LDAS in combination, and so regardless of whether Spivey promises that its disclose may reduce latency by "two orders of magnitude," Petitioner's Ellis/Spivey combination would not do so.

79. For this reason, and the reasons I have stated above regarding altered message flow and the lack of a live data access server, neither Petitioner nor Dr. Wicker has shown how the proposed combination of teachings would actually minimize latency.

B. Ground 2C: Petitioner Fails to Establish that Herzog in View of Spivey Renders Obvious Every Limitation of the Challenged Claims

1. *Petitioner's proposed combination of teachings suffers from many problems.*

80. Notably, Herzog is not necessarily equipped to handle additional, faster-arriving data without further modifications. For example, to the extent that Spivey is capable of providing faster data in the context of the Herzog system, the faster-arriving data would necessarily be processed in consideration for subsequent timestamp synchronization matching with associated video/audio fees, which have a significantly greater latency period. Herzog describes the audio/video as having a substantially longer delay (up to “63 seconds”) as compared to the telemetry data feed delay (up to “2 seconds”). (EX-1015, ¶51.) In this regard, to the extent that Petitioner proposes to speed up with the telemetry data transmission with Spivey message queue server, Petitioner fails to address the question of what Herzog would do with all of the faster arriving data while said data is essentially waiting for the audio/video feed. For example, a POSITA would understand that the import of additional data (because the data arrives faster, as proposed by Petitioner) would impose some data processing requirement on the Herzog system. As such, additional servers, databases, and/or other components would assumingly be required in Petitioner's combination to meet such additional demands brought on the by the faster arriving data. Nevertheless, Petitioner does not suggest such additional

components, and because of this lack of additional components to handle the proposed faster arriving data, the Petitioner's proposed combination is not suited for operation as intended.

81. But even if Petitioner had proposed additional components (e.g., servers, databases) to handle the additional, faster-arriving data, a POSITA would have also understood that with each addition of a computer architecture component comes additional latency. Therefore, a POSITA would not understand that additional latency brought on by the inclusion of additional processing components would improve the Herzog system in any manner. Indeed, the additional latency brought on by any additional processing components would be understood to counteract the latency benefits afforded by Spivey (if any).

2. *A POSITA would not have been motivated to combine the references' teachings.*

82. Fundamentally, the inclusion of the Spivey message queue server device into the Herzog Origin Server would not reduce latency by any meaningful amount. As such, a POSITA would not view the incorporation of such message queue server device into the origin server as an improvement, particularly in view of the additional complications and drawbacks that its inclusion would entail.

83. To illustrate, Herzog describes, and its techniques rely upon, two distinct latency periods. First, Herzog describes a latency period for the metadata, which has a latency of up to "2 seconds." (EX-1015, ¶82.) Second, Herzog

describes a latency period for the audio/video feed, which has a latency of up to “63 seconds.” (*Id.*) Accordingly, Herzog describes that the audio/feed encoding, decoding, and transmission process accounts for and contributes to the substantial majority of the latency. At most, the teachings of Spivey could be applied to address the 2 seconds of latency associated with the metadata. This is because, as I have explained in detail herein, Spivey does not concern the transmission of audio/video. (EX-2001, Sections V.B, VI.A.1.) And regardless, Petitioner proposes to “maintain the video content delivery system” and Herzog. (Petition, 19.) Therefore, the only possible outcome for the Petitioner’s proposal is a reduction of the 2 seconds of latency associated with the metadata, and not a reduction of the 63 second of latency associated with the audio/video feed.

84. As I explain herein, the message queue server device would not be operable to reduce latency in Herzog whatsoever. Notwithstanding, even if the message queue server device were to be used to reduce metadata latency in Herzog, and even if the message queue server device were to reduce 100% of the latency associated with the metadata, this literal impossible reduction in latency would at most reduce the total latency in Herzog to 61 seconds. This is because, the latency cause by the origin server will equally impact the metadata and the video streams. Figure 4 shows that both the meta data from in the ERP server and the video data from the MPEG4 Publishing Encoders need to transit the origin server to reach the

client. Herzog states, “the origin server 54 is a server transfers data to other servers.” (EX-1015, ¶51.) Therefore the maximum amount of latency the origin server can be responsible for is the “less than 2 seconds” (EX-1015, ¶74) that Herzog describes as the latency of the metadata. Herzog explains that the “latency of the video between when the user requests a video and when the video is shown on the display is approximately 63 seconds. This is due to the down linking, encoding, publishing and decoding of the video.” (EX-1015, ¶74.) Thus, the remaining 61 seconds of latency will have to come from the other sources Herzog described, i.e., the “down linking, encoding,” or “decoding of the video.” This is further supported by Herzog’s Figure 6 which clearly states that the Live Encoders (81) creates latency of “>32 sec” and the latency associated with the client APK for the video feed is “>24 sec.” These two sources alone, neither of which would be impacted by Petitioner’s and Dr. Wickers alterations, account for at least 56 seconds of latency. In comparison the metadata ERP Generator is only associated with “<1 sec” of latency and the client APK for the meta data is also only associated with “<1 sec” of latency. A reduction in latency from 63 seconds to 61, or even to just to 56 seconds, is not a meaningful reduction (if it were even physically possible), and it does not represent Spivey’s promised latency reduction of “at least two orders of magnitude shorter than conventional time periods of legacy device.” (EX-1017, ¶96.) Accordingly, a POSITA would not be motivated to make the proposed combination

because regardless of whether the proposed combination even worked, the promised latency reduction would not be understood as significant with the context of the Herzog disclosure.

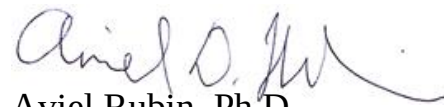
II. CONCLUSION

85. For at least the foregoing reasons, it is my opinion that Petitioner's grounds do not render obvious any of the challenged claims of the '218 patent.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Executed this 6th day of June 2025 in Annapolis, Maryland.

Respectfully submitted,



Aviel Rubin, Ph.D.