

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner

v.

SANDPIPER CDN, LLC,
Patent Owner

Case IPR2025-01010
U.S. Patent No. 10,057,322

**DECLARATION OF SAMUEL H. RUSS, PH.D.,
IN SUPPORT OF THE PATENT OWNER'S RESPONSE**

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

Sandpipper EX2015
Google v. Sandpipper
IPR2025-01010

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	MY BACKGROUND AND QUALIFICATIONS	2
III.	LIST OF DOCUMENTS CONSIDERED	5
IV.	LEGAL UNDERSTANDING	7
	A. My Understanding of Claim Construction.....	7
	B. My Understanding of Obviousness	8
	C. My Understanding of a Person of Ordinary Skill in the Art.....	10
V.	PERSON OF ORDINARY SKILL IN THE ART	11
VI.	OVERVIEW OF THE STATE OF THE ART.....	11
VII.	THE 322 PATENT	13
	A. Overview of the 322 Patent	13
	B. Overview of the Prosecution History of the '322 Patent	16
	C. Claim Construction.....	18
VIII.	OVERVIEW OF THE PRIMARILY RELIED ON REFERENCES.....	22
	A. Verma	22
	B. Raciborski.....	24
	C. Lewin.....	25
IX.	DR. MOWRY'S PROPOSED OBVIOUSNESS GROUNDS.....	26
	A. Dr. Mowry does not establish that the cited art teaches a "direct link" to a CDN node as required by all challenged claims.	26
	1. Verma's "akamaized" URLs are DNS-dependent hostname URLs, not "direct links."	27
	2. Raciborski requires DNS resolution and does not fill Verma's deficiency (Grounds 2, 4, 5).....	32
	3. Lewin also requires DNS resolution and does not fill Verma's deficiency (Grounds 3, 4, 5).....	33
	B. Dr. Mowry does not establish that the cited art renders obvious the "tag" limitations of independent claim 1 (Grounds 3-5).....	34

1.	Lewin’s ESI tags do not provide the claimed “embedded resource with a tag.”.....	35
2.	Dr. Mowry has not established a motivation to combine Verma and Lewin.....	38
3.	Dr. Mowry does not explain how tag identification in the combined system triggers the origin system’s use of delivery parameters to obtain a modified embedded resource.	42
C.	Dr. Mowry does not establish that the cited art discloses or renders obvious the claimed “application programming interface (API)” (Grounds 1A/1B, 2, 4; Claims 11 and 15).	44
1.	Claims 2, 9, and 11–15 require an exposed “API” to receive requests, regardless of whether the API includes additional functionality.	45
2.	Dr. Mowry has not shown that Verma’s broker provides the claimed “API.”	51
X.	CONCLUSION.....	57

I, Samuel H. Russ, hereby declare as follows.

I. INTRODUCTION

1. I have been retained as an expert witness on behalf of Patent Owner Sandpiper CDN, LLC (“Patent Owner”) for the above-captioned *inter partes* review (“IPR”). I am being compensated for my time in connection with this IPR at my normal hourly rate. My compensation is not contingent on either my expert opinion or the outcome of this proceeding. I have no other interest in this proceeding.

2. I understand that this declaration accompanies Patent Owner’s Response filed in the IPR involving U.S. Patent No. 10,057,322 (“the ’322 patent”) (EX1001), which resulted from U.S. Application 14/985,968 (“the ’968 application”), filed December 31, 2015, which claims priority to U.S. Provisional Application No. 62/098,930, filed December 31, 2014. My opinions regarding the knowledge in the art at the time of the ’322 patent are based on this 2014–2015 timeframe.

3. In preparing this declaration, I have reviewed the ’322 patent and each of the documents cited herein, in light of general knowledge in the art before the time of the ’322 patent. In formulating my opinions, I have relied upon my experience, education, and knowledge in the relevant art. In formulating my opinions, I have also considered the viewpoint of a person of ordinary skill in the

art (“POSITA”) (as discussed further below in Section V) at the time of the ’322 patent.

II. MY BACKGROUND AND QUALIFICATIONS

4. I received a Bachelor’s degree in Electrical Engineering from the Georgia Institute of Technology (“Georgia Tech”) in 1986 and a Ph.D. in Electrical Engineering from Georgia Tech in 1991.

5. From 2007 to the present, I have been a member of the faculty of the University of South Alabama as an Assistant and Associate Professor in the Department of Electrical and Computer Engineering. During that time, I have won awards for excellent teaching and have been actively publishing research in home networking and digital video recording (DVR) technologies. I am active in the Institute of Electrical and Electronic Engineers (IEEE) and was a Distinguished Lecturer for the IEEE Consumer Electronics Society. As an example of the type of work I perform at South Alabama, I managed the design, construction, and launch of a fully functional satellite from the International Space Station into space in 2022.

6. From 2000 to 2007, I worked for Scientific-Atlanta (now Cisco’s Service Provider Video Tech. Group), where I managed a cable set-top box (STB) design group that designed four STB models, including the Explorer 4200 (non-DVR) and 8300 (DVR) models. Both models sold several million units. As design-

group manager, I was responsible for managing the design and prototyping activities of the group and for interfacing with other groups (especially integrated-circuit design, procurement, software developers, the factory where prototypes were built, and product managers) and for maintaining the hardware and mechanical development schedule. Since the products were produced in extremely high volumes, the projects had very high visibility in the company, and therefore carried a great deal of responsibility.

7. Also while at Scientific-Atlanta, I became a staff expert in home networking, conducting demonstrations of wireless video technology and managing a group that developed a new coaxial home networking system. The coaxial system won a Technology and Engineering Emmy® Award in 2013. I became a staff expert in DVR reliability, and led a team that improved the software, hardware, repair, and manufacturing processes. I am a named inventor on fifty-one (51) patent applications that were filed while I was at Scientific-Atlanta, twenty-eight (28) of which have issued as U.S. patents as of the writing of this report.

8. Both my work with home networking and coaxial DVR video distribution were technologies that improve content distribution networks (CDNs). In order to implement these systems, I had to develop a detailed understanding of, *e.g.*, bandwidth and resource management, video distribution, and the requirements

for storage media to support multiple video streams. For example, because of my work on multi-room DVR and hard drive reliability, I identified the level of performance needed for a hard disk drive to support seven simultaneous streams of video when functioning as a home video server. The set-top boxes I designed supported video on demand and switched digital video, both forms of content distribution.

9. From 1999 to 2000, I was a Staff Electrical Engineer and then Matrix Manager at IVI Checkmate (now Ingenico), where I managed the hardware design team that completed the design of the eN-Touch 1000 payment terminal. This terminal was in widespread use, for example, at the self-checkout at Home Depot.

10. I also served on the faculty of Mississippi State University from 1994 to 1999 as an Assistant Professor in the Department of Electrical & Computer Engineering where I taught circuit board design and two-way interactive video classes, among other things.

11. I have also authored 32 journal articles and conference papers. A conference paper on digital video recording won second place in a “best paper” competition at the 2011 International Conference on Consumer Electronics in Las Vegas, NV.

12. I understand that a copy of my complete *curriculum vitae* has been submitted as Exhibit 2011, which includes a list of my publications and the cases

in which I have testified as an expert in deposition or trial in the past four years,
and other details regarding my expertise and qualifications.

III. LIST OF DOCUMENTS CONSIDERED

13. I have considered the following documents listed below in addition to any other documents cited in this declaration. To the best of my knowledge, these documents are true and accurate copies of what they purport to be. An expert in the field would reasonably rely on them to formulate opinions such as those set forth in this declaration.

Paper No.	Description
1	Petition for <i>Inter Partes</i> Review of U.S. Patent No. 10,057,322
15	Institution Decision

Exhibit No.	Description
1001	U.S. Patent No. 10,057,322 (“the ’322 patent”)
1002	Prosecution History of the ’322 patent
1003	Declaration of Todd C. Mowry, Ph.D.
1004	<i>Curriculum Vitae</i> of Todd C. Mowry, Ph.D.
1005	U.S. Patent Application Publication No. 2009/0132640 to Verma (“Verma”)
1006	U.S. Patent Application Publication No. 2007/0067424 to Raciborski (“Raciborski”)

Exhibit No.	Description
1007	U.S. Patent Application Publication No. 2014/0052811 to Lewin (“Lewin”)
1008	U.S. Patent Application Publication No. 2009/0172167 to Draï (“Draï”)
1009	U.S. Patent No. 6,108,703 to Leighton (“Leighton”)
1010	U.S. Patent No. 8,364,541 to Roth (“Roth”)
1011	U.S. Patent No. 5,835,712 to DuFresne (“DuFresne”)
1012	U.S. Patent No. 6,314,439 to Bates (“Bates”)
1013	U.S. Patent No. 7,210,099 to Rohrabough (“Rohrabough”)
1014	U.S. Patent No. 7,299,291 to Shaw (“Shaw”)
1015	U.S. Patent No. 6,542,964 to Scharber (“Scharber”)
1016	U.S. Patent No. 6,119,155 to Rossmann (“Rossmann”)
2001	“Cloud CDN: content delivery network,” <i>Google Cloud</i> , accessed at https://cloud.google.com/cdn (last accessed July 2, 2025)
2012	Goodwin, M., “What is an API (application programming interface)?” accessed at https://www.ibm.com/think/topics/api (last accessed August 5, 2025)
2013	“CDN Stands For Content Delivery Network: History and Evolution,” <i>BlazingCDN</i> , accessed at https://blog.blazingcdn.com/en-us/cdn-stands-for-content-delivery-network-history-and-evolution (last accessed August 1, 2025)
2014	“What is a CDN (Content Delivery Network)?” <i>Amazon Web Services</i> , accessed at https://aws.amazon.com/what-is/cdn/ (last accessed August 1, 2025)

Exhibit No.	Description
2016	Deposition Transcript of Todd C. Mowry
2017	“ARL syntax,” Akamai Techdocs, accessed at https://techdocs.akamai.com/edge-diagnostics/docs/arl-syntax (last accessed February 3, 2026)
2018	Cohen, E. & Kaplan, H., “The Age Penalty and its Effect on Cache Performance,” <i>3rd USENIX Symposium on Internet Technologies and Systems</i> (2001)
2019	“What Is an API (Application Programming Interface)?” Oracle, available at https://www.oracle.com/cloud/cloud-native/api-management/what-is-api/ (last accessed March 3, 2026)
2020	“What’s the difference between APIs and a simple http request?,” Medium, available at https://medium.com/@paul.k.pallaghy/whats-the-difference-between-apis-and-a-simple-http-request-c31692355dd1

IV. LEGAL UNDERSTANDING

A. My Understanding of Claim Construction

14. I understand that during an *inter partes* review proceeding, claims are to be construed in light of the specification as would be read by a person of ordinary skill in the relevant art at the time of the invention. I understand that claim terms are given their plain and ordinary meaning as would be understood by a person of ordinary skill in the relevant art in the context of the patent’s specification and prosecution history. A claim term, however, will not receive its ordinary meaning if the patentee acted as his or her own lexicographer and clearly

set forth a definition of the claim term in the specification. In this case, the claim term will receive the definition set forth in the patent.

B. My Understanding of Obviousness

15. I understand that a patent claim is invalid if the claimed invention would have been obvious to a person of ordinary skill in the art at the time the application was filed. I understand that this means that even if all of the requirements of the claim cannot be found in a single prior-art reference that would anticipate the claim, the claim can still be invalid.

16. To obtain a patent, a claimed invention must have been, as of its priority date, nonobvious in view of the prior art in the field. I understand that a patent claim is obvious when the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art.

17. I understand that to prove that prior art or a combination of prior art renders a patent claim obvious, it is necessary to: (1) identify the particular references that, singly or in combination, render the patent claim obvious; (2) specifically identify which elements of the patent claim appear in each of the asserted references; and (3) explain how the prior art references could have been combined in order to create the inventions claimed in the patent claim.

18. I also understand that when considering the obviousness of a patent claim, one should consider whether a teaching, suggestion, or motivation to combine the references exists so as to avoid impermissibly applying hindsight when considering the prior art. I understand this test should not be rigidly applied, but that the test can be important to avoiding such hindsight.

19. I understand that certain objective indicia can be important evidence as to whether a patent is obvious or nonobvious. Such indicia include:

(1) commercial success of products covered by the patent claims; (2) a long-felt need for the invention; (3) failed attempts by others to make the invention; (4) copying of the invention by others in the field; (5) unexpected results achieved by the invention as compared to the closest prior art; (6) praise of the invention by the infringer or others in the field; (7) the taking of licenses under the patent by others; (8) expressions of surprise by experts and those skilled in the art at the making of the invention; and (9) the patentee proceeded contrary to the accepted wisdom of the prior art.

20. I also understand that “obviousness” is a legal conclusion based on the underlying factual issues of the scope and content of the prior art, the differences between the claimed invention and the prior art, the level of ordinary skill in the pertinent art, and any objective indicia of non-obviousness. For that reason, I am not rendering a legal opinion on the ultimate legal question of obviousness. Rather,

my testimony addresses the underlying facts and factual analysis that would support a legal conclusion of obviousness or non-obviousness, and when I use the term obvious, I am referring to the perspective of one of ordinary skill at the time of the '322 patent.

C. My Understanding of a Person of Ordinary Skill in the Art

21. I understand that a POSITA is presumed to be aware of all pertinent art, thinks along conventional wisdom in the art, and is a person of ordinary creativity.

22. I understand that the following should be considered when deciding the level of ordinary skill in the art that someone would have had at the time the claimed invention was made:

- the levels of education and experience of persons working in the field;
- the types of problems encountered in the field; and
- the sophistication of the technology.

23. My opinion below explains how a POSITA would have understood the technology described in the documents I have identified herein around the time of the '322 patent.

24. Regardless of whether I use “I” or “POSITA” during my technical analysis below, all of my statements and opinions are always to be understood to be based on how a POSITA would have understood or read a document at the time

of the '322 patent.

V. PERSON OF ORDINARY SKILL IN THE ART

25. In my opinion, a POSITA at the time of the '322 patent would have had at least a bachelor's degree in computer science, electrical engineering, or other suitable field, and at least two years of work or research experience related to the field of content delivery networks. Further professional experience could substitute for education, and vice versa. It is also my opinion that the definition of a POSITA would not materially affect the analysis required for upholding the patentability of the challenged claims.

VI. OVERVIEW OF THE STATE OF THE ART

26. Content delivery networks (CDNs) provide critical services that enable content providers to quickly deliver online content to millions of consumers simultaneously over the Internet. In the early 1990s, the World Wide Web saw increasing adoption, becoming a household staple. This mass adoption led to data congestion issues from the ever-growing number of users seeking to simultaneously access Internet content. EX2013, 2.

27. A typical computer server in the 1990s could only handle a limited number of simultaneous connections before becoming overloaded. Moreover, signals take time to traverse routers and move through content-congested internet cables. Consumers living far from the physical server(s) hosting content

experienced sluggish load times and high latency due to problems such as overloaded servers, congested network segments, and geographic separation. EX2013, 2-3.

28. The availability of CDN servers around the world allowed service providers to replicate appropriate content from customers' origin servers to appropriate CDN servers, and transparently rendezvous end users requesting that content to the "best" CDN server to deliver that content, all while providing their customers with control over their content and user experience. EX2013, 3-6; EX2014, 2-3. This service and its architecture were quickly adopted by many others in the industry. EX2013, 3-6; EX2014, 2-3.

29. In the CDN model, duplicate copies of content are stored in servers located over a wide geographic area (*e.g.*, across the country or around the world). EX2013, 2; EX2014, 2-3. Propagating data from origin servers to edge servers is known as "caching." EX2013, 3-6; EX2014, 2-3. In other words, when data is propagated from its origin to one of the geographically distributed CDN servers, the data is cached at the CDN server. EX2013, 3-6; EX2014, 2-3. Using a CDN, end-users connect to an optimal edge server—one that that is closer to them or that has available capacity. EX2014, 2-3. Distributing content across a network of servers alleviated data congestion issues and allowing consumers to connect to edge servers located near them reduced latency. Importantly, the CDN model is

invisible to the user—they simply see the content they requested. Note that the closest server may not necessarily refer to one that is geographically closest but rather has the lowest latency. EX2013, 3; EX2014, 1.

30. In the early-to-mid 2000s, demand for CDNs exploded. EX2013, 7-8. Content providers like Google and YouTube use CDN technology to meet their enormous data streaming needs, as well as for video search results. EX2001, 1-3. As a result, techniques for controlling how users access content on a CDN became increasingly important.

VII. THE 322 PATENT

A. Overview of the 322 Patent

31. The '322 is directed to techniques for improving the performance of content delivery networks (CDNs), in some cases using an application programming interface (API) to facilitate embedding of “link resolution information within an HTML document or other content returned to a requesting client.” EX1001, 1:13-18. In particular, aspects of the '322 patent “involve an alternative to using conventional domain name system (DNS) servers for edge selection by instead calling an API or otherwise querying the CDN when constructing a uniform resource locator (URL) for the resource to be delivered from the CDN.” *Id.*, 3:9-14.

32. In conventional DNS flows, a “client-side” DNS resolver “contacts

(e.g., queries) a DNS infrastructure 16, which may involve many different components, to resolve a network address for a resource.” EX1001, 3:64–4:4. For example, “when a user enters, into the browser Web page, an address `http://www.example.com`,” the browser “contacts the resolver 14, which in turn requests the internet protocol (IP) address for the Web page from DNS.” *Id.*, 4:5–9. In this example, “`www.example.com`” is a hostname that is resolved via conventional DNS lookup. As a result, “DNS returns the IP address of an origin server 18 capable of delivering a hypertext markup language (HTML) document 20 that will render the Web page in the browser. *Id.*, 4:9–12. Following these conventional flows, users were able to access more content (e.g., television, movies, news, software updates) on networks. *Id.*, 1:22–26, 1:39–50.

33. The inventors of the ’322 patent, however, realized that following the conventional flow to “resolve a network address” using “many different components” introduced technical inefficiencies, and thus devised unconventional techniques that involved “bypassing DNS,” which would reduce the number of components involved in servicing requests while improving response speeds when delivering content. EX1001, 2:45-49, 4:1-4, 5:29-32, 6:60-7:1. They “[n]ote” this is accomplished, for example, by “embedding an IP address to serve the content directly,” which would “connect directly to the source of the embedded resource.” *Id.*, 5:29-32.

34. These techniques are recited in the claims of the '322 patent. For instance, independent claim 15 recites:

A method comprising:

providing, at a content delivery network, an application programming interface (API);

at the API, receiving, from an origin system that is separate from the content delivery network, an embedded resource and at least one parameter associated with the embedded resource;

generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource;

delivering, by the content delivery network, the modified embedded resource to the origin system;

receiving, at the node, a request from a client device to obtain the content; and

providing the content to the client device.

EX1001, 12:5-20.

By computing “an address that is optimally suited to serve the content to the requesting client” and “connect[ing] directly to the source of the embedded resource,” the solutions described and claimed in the '322 patent improve the functioning of a CDN, including, for example, offloading computational resource consumption away from a client-side DNS infrastructure, and instead having content retrieved from an optimal node in the network. EX1001, 4:55-56, 5:29-32;

see also id., 4:66-67 (describing additional technical optimizations around “cost, latency, quality, optimal CDN use, and the like”).

B. Overview of the Prosecution History of the '322 Patent

35. I understand that the '322 patent was filed as U.S. Application No. 14/985,968 (“the '968 application”) on December 31, 2015. I also understand that the '322 patent claims priority to U.S. Provisional Application No. 62/098,930, filed December 31, 2014. I understand that the '322 patent received a thorough prosecution on the merits, including overcoming both §§ 101 and 102 rejections. EX1002, 15-18, 60-74, 95-108.

36. I also understand that after receiving a non-final rejection, *see* EX1002, 95-108, Patent Owner amended the independent claims to clarify the architecture of the claimed systems and methods. EX1002, 56-59; For example, Patent Owner amended independent claim 12 (which issued as independent claim 11) as follows:

12. (Currently Amended) A content delivery ~~device~~ network comprising:
at least one processor in communication with a non-transitory computer readable medium
including computer readable instructions for:
~~generating~~ providing an application programming interface (API);
at the API, receiving, from an origin system that is separate from the content
delivery network, an embedded resource and at least one parameter associated with the
embedded resource;
generating a modified embedded resource providing ~~information as to a location~~
~~within a~~ a direct link to a node within the content delivery network for obtaining content
associated with the embedded resource;
delivering, by the content delivery network, the modified embedded resource to
the origin system;
receiving, at the node, a request from a client device to obtain the content; and
providing the content to the client device.

EX1002, 57-58.

37. Patent Owner also added independent claim 22 (which issued as
independent claim 15) with similar limitations:

22. (New) A method comprising:
providing, at a content delivery network, an application programming interface (API);
at the API, receiving, from an origin system that is separate from the content delivery
network, an embedded resource and at least one parameter associated with the embedded
resource;
generating a modified embedded resource providing a direct link to a node within the
content delivery network for obtaining content associated with the embedded resource;
delivering, by the content delivery network, the modified embedded resource to the origin
system;
receiving, at the node, a request from a client device to obtain the content; and
providing the content to the client device.

EX1002, 59.

38. I understand that Patent Owner argued that the claimed architecture differed from the prior art (U.S. Publication No. 2014/0289319 to Richardson) because “Richardson require[d] the client to either (a) communicate with the DNS server of the CDN to determine the node that will be used to deliver the content; or (b) execute translation code to ask the CDN for translation information, which translation information may include the address of the node that will be used to deliver the content.” EX1002, 72-73. Richardson thus did not disclose “generating [a] modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource.” *Id.* Upon consideration of this and other arguments, I understand that the Examiner allowed the claims. EX1002, 15-18.

39. As I discuss below, it is my opinion that the prior art presented in the Petition also fails to disclose or render obvious the limitations of the independent claims found allowable by the Examiner.

C. Claim Construction

40. I have been asked to opine on how a POSITA would have interpreted the term “direct link” in the context of the ’322 patent. I have applied the plain and ordinary meaning of all other terms as would be understood by a POSITA in the context of the patent’s specification and prosecution history.

41. Each independent claim of the ’322 patent recites a “modified

embedded resource” that includes or provides “a *direct link* to a node within the content delivery network.” EX1001, 10:5–18 (Claim 1) (“wherein the modified embedded resource includes a *direct link*¹ to a node within the content delivery network that can deliver content associated with the embedded resource”), 10:59–11:9 (Claim 11) (“generating a modified embedded resource providing a *direct link* to a node within the content delivery network for obtaining content associated with the embedded resource”), 12:5–20 (Claim 15) (“generating a modified embedded resource providing a *direct link* to a node within the content delivery network for obtaining content associated with the embedded resource”). The phrase “direct link” is not a term of art. In my opinion, a POSITA would have understood that the term “direct link,” used in the context of the ’322 patent specification and prosecution history, refers to a network address that enables a client device to bypass conventional DNS resolution of a hostname when contacting a CDN node.

42. Specifically, the ’322 patent specification confirms that a “direct link” is one that allows the client device to bypass conventional DNS resolution. For example, Figure 1 of the ’322 patent provides “a network diagram and message flow related to generating an embedded resource for inclusion in a network asset, such as an HTML document, the embedded resource *providing a connection to a*

¹ Emphasis added unless otherwise indicated.

node of a CDN bypassing DNS.” EX1001, 2:47–49. The ’322 patent explains that “[t]he location of the client may be used in selecting an optimal node to serve a request for the embedded resource,” and “by *embedding an IP address* to serve the content directly, *the client device may bypass DNS and instead connect directly to the source of the embedded resource.*” EX1001, 5:27–32. In other words, the specification expressly equates “directly” connecting with bypassing DNS through an embedded IP address. *Id.* Based on this disclosure, it is my opinion that a POSITA would have understood the “direct link” to be one that provides the client with an address enabling it to initiate a connection to the CDN without first performing a conventional DNS lookup.

43. As I explained in Section VII.A, DNS resolution is the standard mechanism by which a hostname is resolved to the IP address required to establish a TCP connection. Resolving a hostname via conventional DNS flows, however, involves “many different components,” such as contacting a resolver, which then “requests the internet protocol (IP) address” for the hostname. EX1001, 4:1–9. The specification notes these qualities and thus offers a key technical improvement that bypasses that process. *Id.*, 5:29–32, 2:45–49. Indeed, the ’322 patent specification premises its disclosure on this improvement, describing that “[a]spects of the present disclosure involve *an alternative to using conventional domain name system (DNS) servers for edge selection.*” EX1001, 3:9–14. Notably,

the Petition also relies on this very sentence to characterize the '322 patent, acknowledging the '322 patent's focus on DNS-bypass technology. Pet., 8.

44. The remaining description provided in the specification further confirms that a “direct link” is one that allows the client device to bypass conventional DNS resolution. Figure 3, for example, includes operation 330 that returns a “network address of [an] identified node” to the origin server for inclusion as the embedded resource. EX1001, FIG. 3 (Operation 330). The associated description provides that “[t]he API returns an absolute URL from which the content may be retrieved,” which may include an “IP address[]” so that “[w]hen the user selects the link, the browser will *directly contact the node* for the content.” *Id.*, 6:50–62. And in the case that the “absolute URL” includes “a hostname,” the patent explains that “the embedded resource may include specific reference to the CDN such that name servers 34 of the CDN are *contacted directly, bypassing any other DNS processing.*” EX1001, 6:50–7:1. In other words, regardless of the form of the “absolute URL” included in the embedded resource, the '322 patent describes that the link allows for bypassing of conventional DNS processing. The CDN is contacted directly in all instances, in line with the '322 patent's stated goal of providing “an alternative to using conventional domain name system (DNS) servers for edge selection.” EX1001, 3:9–14.

45. I understand that Petitioner interprets “direct link” as effectively

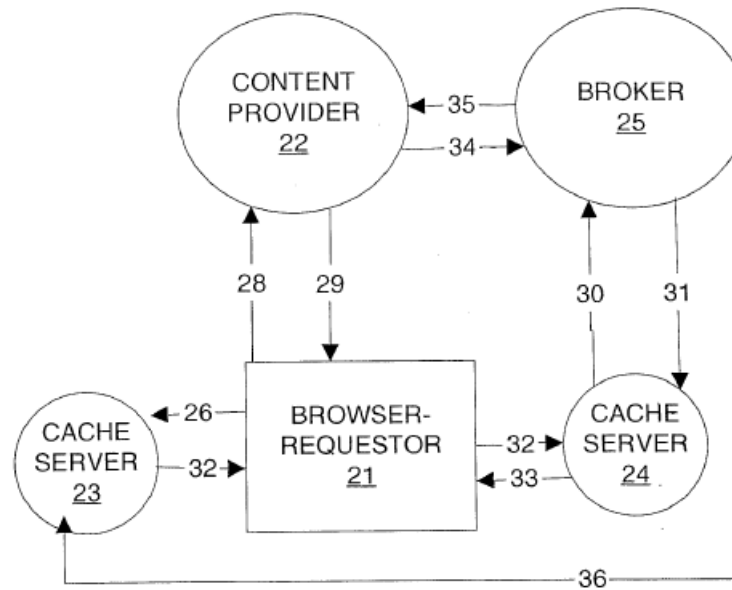
encompassing any complete hostname-based URL—including one requiring conventional DNS resolution. *See* Pet., 23 (asserting that a “direct link” can be met as long as the link provides a “path bound to a single cache server node”). This interpretation, however, ignores the DNS-bypass requirement of a “direct link” described in the ’322 patent specification. The fact that a URL may map to one particular cache server is not enough to satisfy the claimed “direct link”—that interpretation would incorrectly allow a URL to be resolved through the conventional DNS hierarchy and still meet the “direct link” limitation.

46. Therefore, in view of the ’322 specification, it is my opinion that the proper interpretation of “direct link” is a network address that enables a client device to bypass conventional DNS resolution of a hostname when contacting a CDN node.

VIII. OVERVIEW OF THE PRIMARILY RELIED ON REFERENCES

A. Verma

47. Verma discloses a “computer method and related apparatus for delivering content files to a client computer.” EX1005, Abstract, ¶¶10-11. Verma’s method includes modifying a resource link—based on client location and content availability—to direct requests to a cache server node in a CDN. Figure 1 shows an exemplary “content delivery network.”



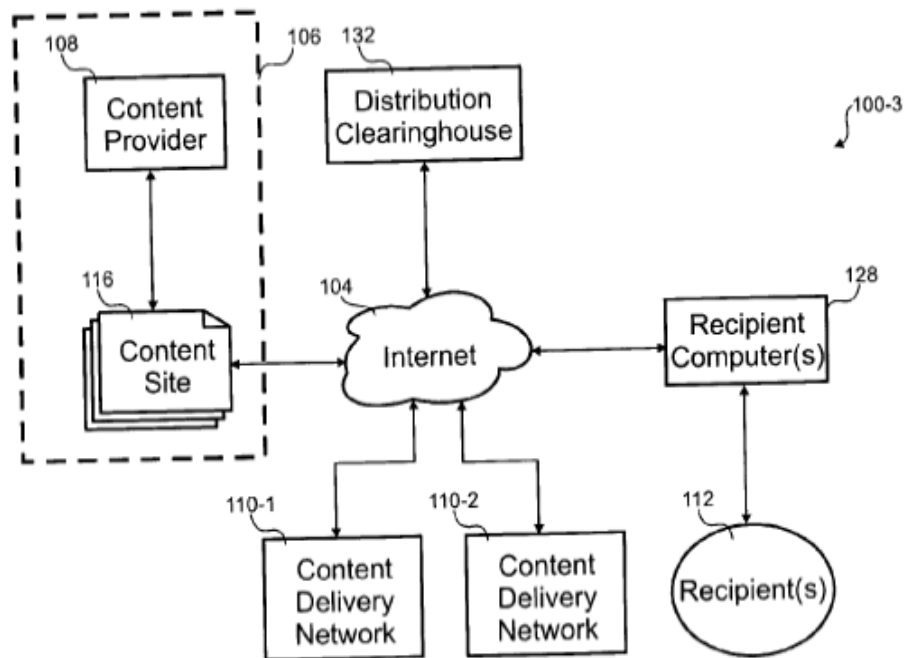
EX1005, FIG. 1.

48. As illustrated in Verma's Figure 1, the content delivery network includes a client that runs a "browser 21," a "content provider 22," a "content delivery network broker 25," and "a set of cache servers" "23"/"24." EX1005, FIG. 1, ¶¶22, 25. The client "browser 21 operates to request a content file ... by sending a URL 28 to a content provider 22." EX1005, ¶25, FIG. 1. "The request from the client ... can comprise delay tolerance information" *Id.*, ¶18. This request is then "forward[ed]" from content provider 22 to the content delivery network broker 25 that "manages at least one, and preferably, a set of cache servers." *Id.*, claim 9, ¶25. Also, content provider 22 can "dynamically request the broker 25 for a content file delivery to a location close to the client." *Id.*, ¶26. The broker sends availability inquiries to the cache servers to "find out whether they

currently store the requested” file. *Id.*, ¶25. The broker 25 can select the appropriate cache server to serve the content and return a “re-written URL to the content server [that] received the [client] request.” *Id.*, ¶¶12, 26-27, 37. When the client receives the prepended (re-written) URL, “it requests the content file from the appropriate cache server when the content is available.” *Id.*, ¶37.

B. Raciborski

49. Raciborski discloses a “content delivery network (CDN) for delivering content objects from a content provider to a recipient.” EX1006, Abstract. Figure 1C shows an exemplary network.



EX1006, FIG. 1C.

50. Within this network, Raciborski discloses a “content provider” that

“sends various information to the one or more CDNs” in the network to facilitate content delivery. EX1006, ¶¶26, 55-56. The information can include the “IP address of the recipient computer,” the “full object location (i.e., server, path and name),” “if delivery can be delayed,” and a “protocol used to deliver the content object,” among other things. *Id.* This information may be included in a “bid message,” like the one illustrated in Table 1. *Id.*, ¶¶55-56, Table 1.

TABLE I

<u>Real Time Bid Message Example</u>	
Field	Example
Bid Identifier	34sd89732k
Recipient IP Address	69.93.18.222
Domain of Content Object	LLNW.com
Path of Content Object	/press/2005releases/
File Name of Content Object	routemark.mp4
Format	MPEG4
Bitrate	200 Kbps
Latency Options	3
QoS Level(s)	3
Packet Loss Options	2
Buffer Duration	24 hours
Max. Volume	1000

EX1006, Table 1.

Such “bid message” parameters, as illustrated in Table 1, provide the CDN with information it can use to select an optimal delivery path. EX1006, Table 1.

C. Lewin

51. Lewin is generally directed to a “dynamic content assembly mechanism” for constructing an “HTML page.” EX1007, ¶¶8, 12, 28, 49. As

Lewin explains, “[t]raditionally,” “directives [] are placed in HTML pages” “that are evaluated on a server before the page is served.” *Id.*, ¶28. These “directives” or “dynamic assembly instructions” may be “URLs for HTML chunks to be incorporated in[to] the final HTML page.” *Id.*, ¶¶49, 58 (“dynamic content assembly page directives”). These “URLs” are “specif[ied]” by “tags” that can be identified when a server “parses the HTML page.” *Id.* As I explain further below, however, Lewin’s “tags” are not part of Lewin’s URLs like they are in the ’322 patent. *Id.*, ¶¶28, 30; EX1001, 4:32–38.

IX. DR. MOWRY’S PROPOSED OBVIOUSNESS GROUNDS

52. For purposes of this declaration, I have been asked to opine on certain positions taken by Petitioner and Dr. Mowry. That I omit certain claims, claim elements, or presented grounds from my analysis should not be viewed as agreement that any claims or claim elements are obvious.

A. Dr. Mowry does not establish that the cited art teaches a “direct link” to a CDN node as required by all challenged claims.

53. Claim 1 recites “wherein the modified embedded resource includes a direct link to a node within the content delivery network that can deliver content associated with the embedded resource.” EX1001, 10:5–18 (Claim 1). Claim 11 recites “generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource.” EX1001, 10:59–11:9 (Claim 11). Claim 15 recites

“generating a modified embedded resource providing a direct link to a node within the content delivery network for obtaining content associated with the embedded resource.” EX1001, 12:5–20 (Claim 15).

54. As I discussed in Section VII.C, a “direct link” in the context of the ’322 patent refers to a network address that enables a client device to bypass conventional DNS resolution of a hostname when contacting a CDN node. As I discuss below, no reference that Dr. Mowry relies upon, alone or in combination, teaches a “direct link.”

1. Verma’s “akamaized” URLs are DNS-dependent hostname URLs, not “direct links.”

55. Dr. Mowry contends that Verma’s “‘prepended URL’ (‘rewritten URL’) provides a direct link to a CDN node for obtaining content associated with the embedded resource, as claimed, because it ‘comprises a link to a content server or a cache server’ allowing a client to directly contact the node to access requested content.” EX1003, ¶121; *see also id.*, ¶119 (opining that Verma’s “rewritten URL” provides “a complete path (direct link) to a ‘cache server’ (node)”). That is, Dr. Mowry contends that a “direct link” refers to any URL that, when accessed, eventually leads to a particular cache server. But this construction reads “direct link” out of the claims because it would be satisfied by *any* URL pointing to a server—including an ordinary hostname-based URL that requires full public DNS resolution.

56. Verma discloses “prepending” (also called “akamaizing”) original content URLs. EX1005, ¶¶13, 26–27. Verma explains that its technique is “to replace the original URL (e.g. `http://contentURL`) with an Akamai-ed URL (e.g. `http://local_cache_server/contentURL`) that redirects to a local cache server.” EX1005, ¶13; *see also id.*, ¶26 (“the content file URLs are prepended (Akamai-ed)”). The resulting URL—for example, “`http://location1/contentURL`” or “`http://location2/contentURL`” (EX1005, ¶29)—substitutes a CDN *hostname* for the original host, a fact which Dr. Mowry himself acknowledges. *See* EX1003, ¶¶49, 149, 151 (Dr. Mowry explaining that each “prepended URL[]” in Verma includes “a hostname for a cache server, ‘location1,’ ‘location2,’ or ‘local_cache_server’”). The resulting URL thus does not contain an IP address or any other form of DNS-bypassing network address. A POSITA would have readily grasped that any text after the protocol marker (after the “//” would necessarily require DNS to resolve. In the cited example, “location2” is a hostname, not an IP address. Since TCP/IP (the protocol on which http is based) requires an actual IP address, DNS must necessarily be used to replace “location2” with an IP address. This is the purpose of DNS – to resolve a hostname to an IP address.

57. This technical distinction is important in the context of the ’322 patent. A generic hostname-based URL (e.g., “`http://edge.cdn.example.net/content.mp4`”) requires the client’s browser to

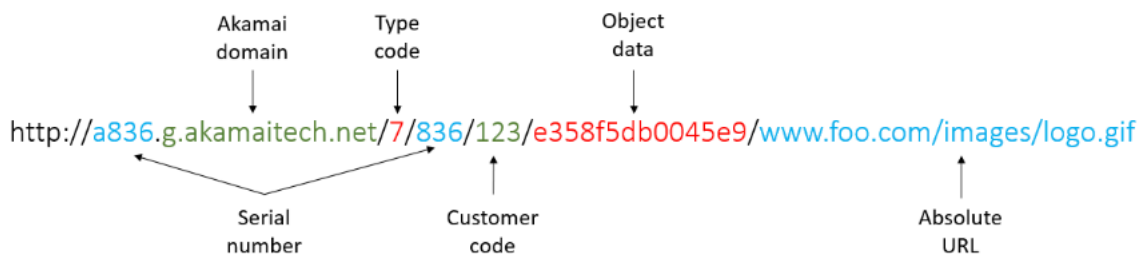
perform a DNS lookup to resolve the hostname to an IP address before any TCP connection can be initiated. *See* EX1001, 3:64–4:12. In contrast, an IP-literal URL (e.g., “http://112.0.2.1/content.mp4”) allows the client to open a TCP connection directly to the specified IP address, bypassing DNS entirely. *See* EX1001, 5:29–32. A POSITA would have readily understood this distinction as fundamental to network addressing, because the public domain name system exists precisely to translate human-readable hostnames into the numerical IP addresses that network protocols require for routing, and whether or not a client must invoke that translation is a basic architectural characteristic of any network addressing scheme.

58. Verma’s akamaized URLs are hostname-based. Every example URL in Verma uses a stand-in hostname (“local_cache_server,” “location1,” “location2”) as the host portion of the URL. EX1005, ¶¶13, 29, 31. The Akamai “URL substitution architecture” used by Verma was well documented when Verma was filed. *See, e.g.,* EX2018, 7–8. When performing “Akamai [] URL substitutions,” “the origin sites substitute the original object URL to one within the CDN domain.” EX2018, 7–8 (substituting “http://cnn.com/images/icons/video.gif” for “http://a388.g.akamaitech.net/7/388/21/e0a3b4215f9e4b/cnn.com/images/icons/video.gif”). “Clients are then directed to a ‘good’ CDN server” such that “[w]hen the client local DNS server resolves the hostname of the object, the Akamai DNS server returns an IP-address of a ‘good’

CDN server.” *Id.*, 8; *see also id.* (discussing “Akamai hostname[s]” including “a388.g.akamaitech.net” and “a534.g.akamaitech.net”).

59. Another example of an akamaized URL (ARL) from Akamai’s technical documentation is depicted below, which is designed to direct users to specific edge serve clusters having “Akamai domain[s]”:

Example of an ARL



EX2017, 1.

In this example, the original content URL is `http://www.foo.com/images/logo.gif`, which gets akamaized/prepended to become `http://a836.g.akamaitech.net/7/836/123/e358f5db0045e9/www.foo.com/images/logo.gif`. *Id.* As a result, instead of resolving the hostname associated with the origin server (`www.foo.com`) via DNS to obtain an IP address to the origin server node that can deliver the content (`logo.gif`), DNS would be used to resolve the hostname associated with the Akamai CDN edge server (`a836.g.akamaitech.net`) to obtain a different IP address to an edge server that can deliver a cached version of the content. *Id.*; *see also* EX1005,

¶¶1–5.

60. Moreover, Dr. Mowry’s own evidence explains that Akamai-style URL rewriting produces a new URL with an altered *hostname* associated with the CDN, for example
`http://ghost467.ghosting.akamai.com/www.provider.com/TECH/images/space.story.gif>space. story.gif.”` EX1003, ¶149 (citing EX1009, 8:2–12). And this rewritten URL requires use of DNS hierarchical processing “to resolve the altered hostname.” EX1009, 9:8–47; EX2016, 55:11-57:8 (explaining that resolution of the hostname “would use the standard techniques that we use with network accesses,” and “[t]he translation of a host name to an IP address is -- on the Internet, is typically done through DNS”).

61. Nowhere does Verma disclose an IP address or any other form of DNS-bypassing address, nor does Dr. Mowry suggest it does. This absence is not incidental; it reflects the fundamental architecture of Verma’s system. Unlike the “direct link” of the ’322 patent, Verma’s entire routing model depends on DNS resolution of CDN hostnames to direct clients to the appropriate edge server. EX1005, ¶¶12–13, 26–27; EX2018, 7–8. And replacing Verma’s hostname-based URLs with static IP-literal addresses would cut against Verma’s load-balancing and content-scheduling architecture as the CDN would no longer be able to leverage DNS-based traffic steering to dynamically route client requests to optimal

edge servers. EX1005, ¶¶12–13, 26–27.

62. Accordingly, I disagree that Dr. Mowry’s contention that Verma’s “‘prepended URL’ (‘rewritten URL’) provides a direct link to a CDN node for obtaining content associated with the embedded resource.” EX1003, ¶¶119–121.

2. Raciborski requires DNS resolution and does not fill Verma’s deficiency (Grounds 2, 4, 5)

63. I understand that Dr. Mowry does not rely on Raciborski to supply either the “modified embedded resource” or the “direct link” limitation of any claim. Raciborski’s role is limited to the parameter-passing and API limitations. EX1003, ¶¶152–186. To the extent Dr. Mowry argues otherwise, Raciborski also fails to disclose a “direct link.” Raciborski’s system also delivers content objects using a CDN hostname—specifically a hostname including the “LLNW.com” domain name. EX1006, ¶¶55, 61, Table I, Table III. Like Verma, this hostname requires DNS resolution; it does not serve as a DNS-bypassing direct link. In particular, a DNS server would need to resolve a LLNW.com hostname to obtain the IP address of the server to obtain the content from. Thus, even assuming that Raciborski could be combined with Verma, the resulting combination would not produce a DNS-bypassing “direct link” because neither reference discloses or suggests returning a DNS-bypassing address as the host portion of a modified embedded resource URL. Adding Raciborski’s LLNW.com hostname URLs to Verma’s akamaized hostname URLs would simply yield a combined system that

still depends entirely on conventional DNS resolution—the opposite of what the ’322 patent claims.

3. Lewin also requires DNS resolution and does not fill Verma’s deficiency (Grounds 3, 4, 5).

64. Dr. Mowry relies on Lewin for the “tag” limitation in Ground 3 and subsequent grounds incorporating Lewin. EX1003, ¶¶187–252. But to the extent Dr. Mowry argues otherwise, Lewin’s system is itself DNS-dependent, and from my review, any alleged combination of Verma and Lewin does not produce a DNS-bypassing direct link.

65. Lewin’s own disclosure confirms that DNS resolution is required. Lewin describes the conventional DNS resolution sequence where “an end user enters a URL into his or her browser, e.g. <http://www.cp.com/index.html>. The browser makes a DNS request to resolve www.cp.com and receives back an IP address for a given ‘edge’ server in the ICDN.” EX1007, ¶49. Lewin’s edge servers and the HTML resources they serve are thus reached only after the client performs a full public DNS lookup. *Id.* This architecture is the same DNS-dependent addressing architecture that the ’322 patent’s “direct link” is designed to bypass.

66. Moreover, the content fragments assembled through Lewin’s ESI tag processing do not constitute DNS-bypassing resources. Lewin’s ESI tags (e.g., `<esi:include>`) are assembly directives that instruct an edge server to incorporate

additional HTML content fragments into a page. EX1007, ¶¶28, 30, 49. Those fragments are retrieved from the same DNS-resolved edge server infrastructure, and at no point in Lewin’s workflow is an IP-literal or otherwise DNS-bypassing address generated. EX1007, ¶49. Even assuming that Lewin could be combined with Verma, the resulting system’s URLs would remain hostname-based—as both references use CDN hostnames—and still require DNS. Neither reference, individually or combined, discloses a mechanism for generating a DNS-bypassing link as required by the ’322 patent independent claims.

67. Therefore, every reference Dr. Mowry relied on for the independent claims (Verma, Raciborski, and Lewin) employs hostname-based addressing that requires conventional DNS resolution—not a “direct link” as the claims recite.

B. Dr. Mowry does not establish that the cited art renders obvious the “tag” limitations of independent claim 1 (Grounds 3–5).

68. Independent claim 1 recites, “receiving, by an origin system, a request from a client device for a content resource, the content resource including *at least one embedded resource with a tag*,” and “*upon identification of the tag*, using at least one delivery parameter to *obtain, by the origin system, a modified embedded resource* from a content delivery network that is separate from the origin system.” EX1001, 10:5–12 (elements [1a] and [1b]).

69. In my opinion, Dr. Mowry does not establish that Verma, in combination with Lewin, renders obvious these limitations for three reasons. First,

Lewin does not disclose an “embedded resource with a tag,” as required by independent claim 1. Second, Dr. Mowry does not establish a motivation to combine Verma and Lewin. Third, Dr. Mowry does not establish a reasonable expectation of success in achieving the combination. Further, even if the references could be combined, the resulting system would still lack a “direct link” for the reasons I set forth in Section IX.A, *supra*.

1. Lewin’s ESI tags do not provide the claimed “embedded resource with a tag.”

70. Dr. Mowry acknowledges that “Verma does not expressly disclose its origin system receiving ‘a request from a client device for a content resource, the content resource including at least one embedded resource with a tag,’” instead relying on Lewin for the claimed “embedded resource with a tag.” EX1003, ¶¶192–202. Specifically, Dr. Mowry asserts that Lewin’s “‘URLs’ are embedded resources,” and these URLs are “‘specif[ied] by ‘tags’ like a ‘<esi:include> tag.’” EX1003, ¶¶197–198 (citing EX1007, ¶¶28, 30, 49). But Lewin does not provide the “embedded resource with a tag.” Contrary to Dr. Mowry’s position and unlike in the ’322 patent, Lewin’s purported “embedded resource” (e.g., a URL) does not have a tag; Lewin’s “tag” is not part of the “embedded resource” at all.

71. Specifically, claim 1 of the ’322 patent requires “at least one *embedded resource with a tag*.” EX1001, 10:6–8. The ’322 patent specification describes this structure, explaining that “[t]he *embedded resource* may include a

tag indicating that a resource address compute engine 30 should be contacted.” *Id.*, 4:32–34. The specification further illustrates the form of the claimed “embedded resource with a tag”: “http://\${CALLAPI:ADDRONLY}/path/resource.txt, or \${CALLAPI:F’ULLURL, path=\$ENV{PATH}, pop=\$POP}.” EX1001, 4:34–48. As illustrated, the tag—the “\${}” construct—is embedded inside or is the URL string itself (i.e., an “embedded resource *with* a tag). Said differently, the tag occupies the position within the address where the CDN node address will go, and the URL cannot be used as-is because the tag is a placeholder that must be resolved.

72. Lewin, by itself or combined with Verma, does not disclose the claimed “embedded resource with a tag.” EX1001, 10:6–8. This is because Lewin’s ESI tags are *not* part of the purported “embedded resource” identified by Dr. Mowry. Instead, they have the inverse structural relationship to their URLs. As exemplified in Lewin’s Figure 6, the relevant structure is: “<esi:include src=us_ad.html/>,” where the “<esi:include>” element is an HTML document-level tag, and the URL (“us_ad.html”) is a value carried as an attribute (“src=”) within the tag. EX1007, FIG. 6, ¶¶28, 30. In Lewin, the URL is inside the tag—not the other way around. Thus, Lewin’s tag is not a marker within the URL indicating that the URL address needs to be computed and replaced; instead, it is merely an HTML assembly directive that happens to reference a URL as one of its

parameters.

73. This inverted structure reflects a fundamentally different functional design. In the '322 patent, the tag signals to the origin server that the URL itself is incomplete and must be resolved. EX1001, 4:40–48. Yet, in Lewin, the tag signals to the edge server that a content fragment should be fetched from the complete URL contained in the tag's attribute and spliced into the HTML page. EX1007, ¶¶28–30, 49. In other words, the opposite manner of operating is reflective of a more fundamental functional difference in how the two references (the '322 patent and Lewin) operate. The structural difference—tag-in-URL vs. URL-in-tag—reflects that fundamental functional difference. Accordingly, it is my opinion that the combination of Verma and Lewin does not teach an “embedded resource *with* a tag,” as claim 1 requires.

74. Additionally, claim 1 requires, “upon identification of the tag, ... obtain[ing], *by the origin system*, a modified embedded resource from a content delivery network that is separate from the origin system.” EX1001, 10:9–12. The “<esi:include>” tag relied upon by Dr. Mowry (EX1003, ¶198) is an assembly directive parsed by a CDN edge server—not by an origin server—to incorporate additional HTML content fragments into a page. EX1007, ¶¶28 (“[T]he functionality is a variant of server side include processing that is sometimes referred to as an ‘edge side include’ to emphasize that the processing is carried out

on an edge server.”), 30, 49 (“[T]he edge server parses the HTML page looking for tags that specify dynamic assembly instructions including, without limitation, URLs for HTML chunks to be incorporated in the final HTML page.”). In particular, Lewin expressly describes its mechanism as “server side include processing” that is “carried out on an edge server.” EX1007, ¶28. The ESI tags instruct the edge server to fetch and assemble pre-existing content blocks (e.g., advertisements, personalized greetings, navigation elements) into a container HTML page at the time of delivery. EX1007, ¶¶28–30, 42, 49. Put simply, Lewin discloses HTML content assembly at an edge server, not address computation or replacement initiated by an origin server.

75. For at least these reasons, it is my opinion that Dr. Mowry fails to demonstrate that the cited art renders obvious the “tag” limitations of claim 1.

2. Dr. Mowry has not established a motivation to combine Verma and Lewin.

76. Even if Lewin taught the claimed “tag” limitations of claim 1 (which it does not in my opinion), a POSITA would not have been motivated to combine Verma and Lewin in the manner Dr. Mowry proposes. Dr. Mowry’s stated motivation for his Verma-Lewin combination is that adding Lewin’s ESI tag-parsing to Verma would give Verma’s system “a more efficient and uniform way to identify URLs” and “dynamic identification and execution of the URLs.” EX1003, ¶¶213–217, 225–235. But, in my opinion, Dr. Mowry’s motivation to

combine is deficient for three reasons: (1) Verma already discloses markup for identifying URLs, and Dr. Mowry provides no reason why a POSITA would have sought to modify Verma's existing markup scheme; (2) Verma's content provider does not require markup tags to identify URLs; and (3) combining Lewin's ESI tag-processing architecture with Verma's system would not achieve the claimed invention. I address each of these reasons in turn.

77. First, adding Lewin's ESI tags to Verma's system to "identify URLs" would be purpose-defeating redundancy. As Dr. Mowry recognizes, Verma already includes standard HTML tags (e.g., "a href" tags) to identify URLs. EX1003, ¶223; EX1005, ¶29. The <a href> tag is the standard HTML mechanism for embedding and identifying a URL in a document—that is its precise function, and it performs it in Verma. Layering Lewin's ESI <esi:include src="[URL]"/> tags on top of a system that already uses <a href> for URL identification would not improve URL identification; it would add a second, structurally different tag system to accomplish the same URL-carrying function the first system already performs. Specifically, the combined system would spend processing time and resources to parse for and identify both sets of tags. A POSITA focused on "improv[ing] the delivery of content files to client computers" and "provid[ing] satisfactory delivery rates" would not introduce such redundancy. EX1005, ¶9.

78. Second, Verma's system operates by allowing the client browser to

request a content file directly—i.e., “by sending a URL 28 to a content provider 22” to process. EX1005, ¶25. Dr. Mowry alleges that “[i]t would have been obvious for Verma’s ‘URL 28’ to be embedded in an HTML page” but provides no reason whatsoever to do so. EX1003, ¶212. In my opinion, it would make little sense to require Verma’s “browser 21” to construct an HTML page with URL 28 embedded within the page before sending a request to content provider 22.

EX1005, ¶25. Verma’s content provider 22 does not require the presence of any tag to identify URL 28 because URL 28 is already received directly from browser 21—it is specifically sent as the target of the request. EX1005, ¶25. The operation suggested by Dr. Mowry would add unnecessary and undesired complexity to Verma’s system, which a POSITA focused on “improv[ing] the delivery of content files to client computers” and “provid[ing] satisfactory delivery rates” would not have sought to introduce. EX1005, ¶9.

79. Third, combining Lewin’s ESI tag-processing architecture with Verma’s system would not achieve the claimed invention. Lewin’s ESI tag-processing architecture is fundamentally incompatible with the operational sequence that element [1b] of claim 1 requires, which recites “upon identification of the tag, using at least one delivery parameter to obtain, by the origin system, a modified embedded resource from a content delivery network.” EX1001, 10:8–12. The information and computation flow runs from the origin server to CDN: the

origin server identifies the tag, invokes the CDN's API with delivery parameters, and receives back a modified embedded resource. EX1001, 4:49–52, 6:24–30.

80. In Lewin, that processing direction is reversed. The origin system is the entity that creates the HTML template containing ESI tags by placing `<esi:include>` directives into the page. EX1007, ¶28. But the origin system does not identify or process those tags. Tag identification and processing occurs at the CDN “edge server,” which parses the template “looking for tags that specify dynamic assembly instructions” and fetches the referenced content fragments from the origin to assemble the final page. EX1007, ¶49. In contrast with the claimed approach, the information flow runs from CDN to origin: the CDN edge identifies the ESI tag, contacts the origin to retrieve content fragments, and assembles the page locally. Put simply, Dr. Mowry's combination completely ignores the “*by the origin system*” requirement in the “upon identification of the tag, using at least one delivery parameter to obtain, by the origin system” claim step.

81. Producing a system that satisfies element [1b] using Lewin's ESI architecture would, therefore, require redesigning Lewin's techniques to account for the opposite processing direction. A POSITA would, for example, be required to design an architecture and corresponding protocol so a tag identification scheme meant to be executed server-side on the CDN edge could be efficiently and practically executed on the client origin system. This combination would also run

counter to conventional edge-architecture paradigms, since the point of edge servers is to “offload[] from the origin server a significant amount of CPU processing and memory,” and reduce overhead at the origin server. EX1007, ¶71. Therefore, in my opinion, reversing Lewin’s processing direction would not have been a routine combination of known elements, but rather an architectural redesign that a POSITA would not have sought to pursue or predictably achieved.

82. For at least these reasons, it is my opinion that Dr. Mowry fails to establish a motivation to combine Verma and Lewin.

3. Dr. Mowry does not explain how tag identification in the combined system triggers the origin system’s use of delivery parameters to obtain a modified embedded resource.

83. Element [1b] recites “upon identification of the tag, using at least one delivery parameter to obtain, by the origin system, a modified embedded resource” and imposes a specific functional sequence: (1) the origin system identifies a tag in the embedded resource; (2) that identification triggers the origin system to use delivery parameters; and (3) the origin system uses those parameters to obtain a modified embedded resource from the CDN. EX1001, 10:9–12. However, the combination Dr. Mowry proposes—Verma for the URL-prepend mechanism and Lewin for the concept of tag identification—leaves the connective tissue of this sequence entirely unexplained.

84. For the “upon identification of the tag” limitation, Dr. Mowry

acknowledges that Verma does not expressly disclose obtaining a modified embedded resource “based upon identification of the tag.” EX1003, ¶233. Dr. Mowry instead turns to Lewin’s description of a server that “‘parses’ an HTML page ‘looking for tags’” and “‘evaluate[s]” directives “placed into HTML pages.” *Id.*, ¶234 (citing EX1007, ¶¶28, 49). From this, Dr. Mowry concludes that “a POSITA would have understood the identification of the tag initiated the dynamic assembly, like ‘incorporat[ion of content] into the final HTML page’ and obtaining a modified embedded resource.” *Id.* But this conclusion ignores how, in the proposed combination, tag identification by the origin system would cause the origin system to use Verma’s alleged delivery parameters and invoke Verma’s URL-prepend mechanism. Dr. Mowry does not provide an explanation.

85. Nowhere in Dr. Mowry’s declaration is there any actual explanation of how those two pieces connect in the combined system to satisfy the “upon identification . . . obtain” causation sequence the claim requires. In Verma, the origin system sends request 34 to the broker and receives a prepended URL in return—a process wholly untethered to tag identification, because Verma’s system does not involve parsing any HTML for tags to obtain the URL from the request. EX1005, ¶¶25–26. And in Lewin, tag parsing is performed by a server in the assembly layer to fetch and incorporate content fragments, not by an origin system to obtain a CDN-modified URL using client delivery parameters. EX1007, ¶¶28,

49. Therefore, neither reference (alone or combined) answers the question of how identifying a tag in the embedded resource causes the origin system to invoke Verma's delivery-parameter-driven, URL-rewriting process. This gap undermines Dr. Mowry's conclusion for element [1b].

86. Based on this same gap in Dr. Mowry's analysis, it is my opinion that a POSITA would not have reasonably expected success in implementing Dr. Mowry's proposed combination. Dr. Mowry has not identified any mechanism by which grafting Lewin's tag-identification concept onto Verma's request-driven URL-rewriting system would produce a combined system that actually performs the "upon identification of the tag, using at least one delivery parameter to obtain" sequence. At best, Dr. Mowry has only demonstrated that the components exist in isolation, not that a POSITA would have sought to combine or had a reasonable expectation of success in combining Verma and Lewin.

87. Therefore, for these additional reasons, it is my opinion that Verma and Lewin do not render obvious the "tag" limitations of claim 1.

C. Dr. Mowry does not establish that the cited art discloses or renders obvious the claimed "application programming interface (API)" (Grounds 1A/1B, 2, 4; Claims 11 and 15).

88. Each of claims 11 and 15 requires "providing, at a content delivery network, an application programming interface (API)" and "at the API, receiving, from an origin system that is separate from the content delivery network, an

embedded resource and at least one parameter associated with the embedded resource.” EX1001, 10:63-67 (Claim 11), 12:6-11 (Claim 15). Dependent claims 2 and 9 also similarly require the origin system to pass “a location identifier” or “information based on the client IP address” to “an application programming interface (API) of the content delivery network.” EX1001, 10:23-27 (Claim 2), 10:52-55 (Claim 9).

89. Dr. Mowry relies solely on Verma for the recited “application programming interface (API)” across all Grounds. EX1003, ¶¶86–87. Although Dr. Mowry further relies on Raciborski and Lewin for certain limitations for Grounds 2 and 4, Dr. Mowry does not rely on either reference for the API in Dr. Mowry’s proposed combinations. EX1003, ¶¶154–158, 164–170, 233, 267–270, 290. For the reasons I discuss below, it is my opinion that Dr. Mowry does not establish that Verma’s broker provides the claimed “API.”

1. Claims 2, 9, and 11–15 require an exposed “API” to receive requests, regardless of whether the API includes additional functionality.

90. I understand that in its Institution Decision, the Board’s preliminarily found that the claimed “API need not be separate from the engine executing on the server.” Paper 17, 10 (citing EX1001, 7:22–25). Based on this preliminary finding, the Board determined on the current record that “Verma’s broker fulfills [my] characterization of an API acting as an interface that allows the client to

communicate with an application executing on the server (that which provides availability information to the broker).” Paper 17, 13–14. In my opinion, the Board’s analysis respectfully reads the term “API” out of the claims.

91. First, the Board’s reliance on the ’322 patent specification’s description that “the API may include the functionality of the engine and not necessarily be considered a discrete mechanism from the computing engine functions,” Paper 17, 10 (citing EX1001, 7:22–25), is misplaced. A POSITA would have understood from reading the specification that the claims do not encompass an embodiment in which the recited “API” is collapsed with the underlying engine executing on the server. Second, regardless of whether the claimed “API” encompasses the underlying engine, the requirement of providing a defined application programming interface still exists. Neither Verma nor the Petition identifies one.

92. With respect to the first point, I understand that the Board preliminarily determined that “because the ’322 patent states that the API ‘may include’ the functionality of the engine, Patent Owner’s example does not show that the API and engine must always be entirely separate components.” Paper 17, 10 (referencing EX1001, 7:22–25); *see also* Pet., 12; Paper 15, 3–4 (Petitioner relying on the same passage). In my opinion, the complete passage does not support this broad reading of the claims. Instead, the passage describes an

alternative embodiment. For reference, the full passage is reproduced below:

In an alternative implementation, which may also involve an API and calling the same in order to resolve an optimal node to serve a request, the process of contacting the API and constructing the optimal network address is instead handled by the CDN and does not necessarily involve a customer origin server or other devices outside the CDN. While embodiments are discussed herein as involving an API, it should be noted that an API is a convenient and efficient way for disparate systems to uniformly and predictably pass information to the computing engine and for the computing engine to return information. However, other forms of communicating with and passing information to the computing engine are also possible. Moreover, the API may include the functionality of the engine and not necessarily be considered a discrete mechanism from the computing engine functions.

EX1001, 7:10–25. As I note above, this passage introduces “an alternative implementation.” EX1001, 7:10. A POSITA would have, therefore, recognized that this passage describes a different embodiment from the primary API architecture disclosed elsewhere in the specification.

93. This understanding is also supported by the description of this implementation “not necessarily involv[ing] a customer origin server or other devices outside the CDN.” EX1001, 7:13–14. That is the opposite of what claims 11 and 15 require. Specifically, those claims recite, “at the API, receiving, from an origin system that is separate from the content delivery network, an embedded

resource and at least one parameter associated with the embedded resource.”

EX1001, 10:63–67 (Claim 11), 12:6–11 (Claim 15). A POSITA reading the specification would have therefore recognized that the “not necessarily discrete” language in this passage describes an alternative embodiment in which the CDN handles address resolution internally—not the claimed embodiment in which the origin server calls an external API exposed by the CDN.

94. In my opinion, the API as recited in the claims follows the specification’s primary embodiment, which is depicted in Figures 1 through 3 and described in detail. *See* EX1001, 3:6–7:9. In that embodiment, the API 26 and the computing engine 30 are shown and described as distinct components: “the API 26 provides the interface to a computing engine 30 that uses the information, passed to the engine through the API.” EX1001, 6:24–30. The origin server affirmatively “calls an API in order to communicate with the compute engine 30 to format the address for the embedded resource.” EX1001, 4:49–52. Figure 3 illustrates this architecture, with a distinct operation for the compute engine receiving inputs through the API (Operation 310) and separate operations for the compute engine identifying nodes and the API returning a network address to the origin server (Operations 320–330). EX1001, FIG. 3; 6:24–53. The API in the primary embodiment, and the one captured by the claims, is therefore a defined, distinct interface layer through which the external origin server invokes the CDN’s

computational functionality.

95. The claim language also confirms this reading. As I noted above, the claims require the API to “receiv[e], from an origin system that is separate from the content delivery network,” an embedded resource and parameters, requiring that the API is an interface accessible to an external caller (e.g., the “origin system”). EX1001, 10:63–67. The claims then separately recite “generating a modified embedded resource” without attributing that step to or reciting the API. EX1001, 10:68–11:2 (Claim 11), 12:12–14 (Claim 15). This distinction is consistent with the primary embodiment’s architecture where the API receives the inputs, and the compute engine generates the output. If the claims contemplated the alternative embodiment—where the API and the engine are collapsed into a single internal CDN component—there would be no reason to recite a separate API at all. The claims could simply recite the CDN generating the modified embedded resource. In my opinion, the fact that the claims call out a distinct “API” as the receiving mechanism confirms that the API is a structurally meaningful interface component, not just a label for whatever internal CDN process happens to participate in content delivery.

96. Nevertheless, regardless of whether the ’322 patent’s claimed “API” encompasses the underlying engine, the requirement of providing a defined application programming interface still exists and should not be read out of the

claims. The specification's description that the API may "not necessarily be considered a discrete mechanism from the computing engine functions" would indicate to a POSITA only that the API may be architecturally integrated with the engine's functionality—for instance, implemented in the same software module or server. But it does not remove the requirement that the API provide an interface through which an external caller invokes computational functionality—that is the purpose of an API. In other words, the '322 patent specification may permit implementation flexibility at the server back end while preserving the API's functional role. But there must still be an exposed API for receiving requests, regardless of whether the API is considered to include processing logic performed after receiving a request.

97. The differences between independent claim 1 and independent claims 11 and 15 further emphasize this reading, in my opinion. In particular, claim 1 does not recite an API, but claims 11 and 15 do. A POSITA, therefore, would have understood claim 1 to encompass broader forms of communication between the "origin server" and the "content delivery network" aside from calling an exposed API. EX1001, 10:5–18; *see also id.*, 3:9–14 (the '322 patent disclosing multiple forms of communication, either "calling an API or otherwise querying a CDN."). In my opinion, interpreting claims 1, 11, and 15 to all encompass the same embodiments would make the "API" recited in claims 1, 11, and 15 meaningless.

98. As I discuss below, Dr. Mowry points to nothing in Verma aside from generic communication between two servers, effectively arguing that any device that receives communication from another device would automatically satisfy the claimed “API” limitations. In my opinion, that interpretation reads the term “API” completely out of the claims.

2. Dr. Mowry has not shown that Verma’s broker provides the claimed “API.”

99. Dr. Mowry contends that “a POSITA would have understood Verma’s ‘broker’ is an API because it performs the same logical functions as the API in the ’322 patent.” EX1003, ¶87 (citing EX1001, 4:49–52, 7:16–25). Dr. Mowry’s argument misses the point. Dr. Mowry’s analysis establishes nothing more than Verma’s “broker” may act as an intermediary between Verma’s “content provider” and “cache servers.” Pet., 11–12. But Verma’s broker receipt and transmission of information does not mean that this communication is performed through an exposed API.

100. A POSITA would have understood that Verma’s broker is an internal CDN management and scheduling component. It “manages at least one, and preferably, a set of cache servers,” sends “availability inquiries” to cache servers to “find out whether they currently store the requested” content file, and “schedule[s] the distribution of content files to edge servers.” EX1005, ¶¶22, 25. When content is not immediately available, the broker “negotiate[s] with the content delivery

network” to “determine the start time, end time, and the cache server where the content file will next be available.” EX1005, ¶38. These are the functions of a CDN-internal scheduling and routing component, not of an interface through which an external origin server invokes CDN computational functionality.

101. Importantly, Verma’s content provider does not disclose that its content provider invokes the broker through any type of defined programmatic interface with structured inputs and outputs. Verma simply describes the content provider “send[ing] a request 34 for information” to the broker, and the broker “provid[ing] content current and/or future availability information” back to the content provider. EX1005, ¶¶25–26. Tellingly, Verma does not disclose or discuss the structure of request 34—including what specific information the content provider passes to the broker. *See* EX1005, ¶¶25–26; Paper 6 at 25. I understand that during deposition Dr. Mowry agreed “[t]here is not a detailed description of a specific syntax for -- or format” of “the request that’s sent from content provider 22 to broker 25” in Verma. EX2016, 36:15-37:12.

102. In contrast to Verma’s broker, which is described as nothing more than a traditional server, the API in the ’322 patent exists precisely to provide the origin server with a defined interface that facilitates communication with a separate compute engine in the CDN. EX1001, 6:24–30 (“the API 26 provides the interface to a computing engine 30”). If the origin server could simply forward a request into

the CDN's internal pipeline, as Verma's content provider does with request 34 (EX1005, ¶25), there would be no need for an API at all.

103. Dr. Mowry's failure to explain how Verma's broker provides an API is further emphasized by the '322 patent's distinction between "calling an API or otherwise querying the CDN." EX1001, 3:9–14. In other words, the '322 patent specification contemplates communication with CDN components without calling an API. The fact that Verma's broker receives requests from Verma's content provider does not resolve the question of whether those requests are received by calling an API. And Dr. Mowry's analysis is entirely silent with respect to that question. *See* EX1003, ¶87; Pet., 11–12.

104. Dr. Mowry further opines that Verma's broker "performs the same logical functions as the API in the '322 patent" because it communicates between the content provider and the CDN. EX1003, ¶87. But Dr. Mowry's analysis consists of describing what Verma's broker does in general terms—e.g., receiving a forwarded request, selecting a cache server, returning a rewritten URL—and then labeling it an "API." EX1003, ¶¶87–88. In my opinion, this approach conflates any intermediary component in a data flow with a programmatic interface. For example, a router sits between a client and a server and a load balancer sits between a client and a pool of servers, yet neither would be considered an API. The claims require an exposed interface that receives defined inputs (parameters)

from an external origin system and generates a defined output (a response).

Otherwise, the term API would have no meaning in the claims.

105. I understand that the Board’s preliminary finding on this limitation also relied on my general characterization of an API as “the bridge establishing the connection between” a client and server, concluding that “this function of Verma’s broker fulfills Dr. Russ’s characterization of an API.” Paper 17 at 13. This finding, in my opinion, applied my background definition at too high a level of abstraction. My description of an API as a “bridge” was intended to be a general explanation of what an API is as a concept in the field of computer science. EX2010, ¶50. It was not intended to define the metes and bounds of the API limitation as used in the claims of the ’322 patent.

106. To add more clarity, a POSITA would have understood that an “API” denotes a defined programmatic interface that specifies the inputs a calling system must supply and the output the receiving system will return. Industry-standard technical literature confirms that these defining characteristics include: (1) the interface is accessible to an external caller (here, the origin server) via an “endpoint”—“[s]pecific URLs that define where to send data and requests”); (2) it accepts structured inputs through defined “parameters”—“[s]pecific details needed for the request”; and (3) it returns a defined output in “response”—“[t]he format of the data sent back by the application.” EX2019, 2–3. These characteristics are

defined through an “API specification [that] provides a structured description of what the API will do and exactly how to interact with it.” *Id.*, 3. This is precisely the architecture the ’322 patent describes: the origin server “calls an API” (EX1001, 4:49–52) using defined parameters—the embedded resource, client IP address, popularity, and geographic information—and receives back a defined response, the “network address of identified node.” EX1001, FIG. 3, 6:50–7:1.

107. Dr. Mowry’s interpretation suggests that *any* communication with a server necessarily involves a call to an API. I disagree with that interpretation. An API specifically “defines how a client should make requests, what responses it will get, and how it can use [a] service’s functionality.” EX2020, 1–2. While the API may make “use of HTTP calls,” it “also includes the standards, protocols, and conventions for those HTTP calls”—any HTTP call to a server does not suffice to establish use of an API. *Id.* Indeed, the ’322 patent makes a point to differentiate between “calling an API” and “otherwise querying the CDN” EX1001, 3:9–14. To assume that every server communication amounts to a call to an API would, as I have noted above, make the term meaningless in the claims.

108. In other words, merely receiving and forwarding a request through an internal CDN pipeline—without a defined interface contract, defined parameters, or a defined API response format—is not an API call. As a POSITA would have recognized, a router receives packets and forwards them, and a load balancer

receives requests and distributes them, yet those intermediary functions do not make them APIs. The distinction is whether the external caller invokes a specific, defined interface with a specified parameter-response contract, or whether it simply sends a message into a system's internal pipeline. Verma's broker receives a "request 34" from the content provider (EX1005, ¶¶25–26), but Verma is entirely silent on the structure of request 34, including what interface definition (if any) governs the exchange. EX2016, 36:15-37:12. Verma discloses no defined programmatic interface, and Dr. Mowry does not point to one.

109. For these reasons, it is my opinion that Verma's broker does not provide the claimed "API" of claims 2, 9, and 11–15.

X. CONCLUSION

In signing this declaration, I recognize that the declaration will be filed as evidence in an *inter partes* review before the Patent Trial and Appeal Board of the United States Patent and Trademark Office. I also recognize that I may be subject to cross-examination in the case and that cross-examination will take place within the United States. If cross-examination is required, I will appear for cross-examination within the United States during the time allotted.

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 on this 4th day of March, 2026, in Mobile, AL.



Samuel H. Russ, Ph.D.