UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION,

Petitioner,

v.

SANDPIPER CDN, LLC,

Patent Owner.

Case IPR2026-00174
Patent No. 9,762,692

PETITION FOR *INTER PARTES REVIEW* OF U.S. PATENT NO. 9,762,692

TABLE OF CONTENTS

LIST OF EXHIBITS

Exhibit	Reference
1001	U.S. Patent No. 9,762,692 (“Ex-1001”)
1002	File History of the ’692 patent
1003	Declaration of Mark R. Lanning
1004	Curriculum Vitae of Mark R. Lanning
1005	U.S. Patent App. Pub. No. 2006/0015574 (“Seed”)
1006	U.S. Patent App. Pub. No. 2004/0221019 (“Swildens”)
1007	David Gourley & Brian Totty, *HTTP The Definitive Guide*, O’Reilly (Sept. 2002)
1008	John Dilley, et al., *Globally Distributed Content Delivery*, IEEE Internet Computing, Vol. 6, Issue 5, 50 (Oct. 31, 2002)
1009	Rajkumar Buyya, et al., *Content Delivery Networks*, Springer (2008)
1010	U.S. Patent App. Pub. No. 2002/0065899 A1
1011	U.S. Patent No. 8,180,720 B1
1012	U.S. Patent No. 7,024,466 B2
1013	Jianliang Xu, et al., *Caching & Prefetching for Web Content Distribution*, IEEE Computing in Sci. & Eng’g, Vol. 6, Issue 4, 54 (Aug. 31, 2004)
1014	Opening Claim Construction Brief, *Sandpiper CDN, LLC v. Comcast Cable Commc’ns, LLC*, No. 2:24-cv-00886 (E.D. Tex. Nov. 14, 2025), ECF No. 88
1015	Responsive Claim Construction Brief, *Sandpiper CDN, LLC v. Comcast Cable Commc’ns, LLC*, No. 2:24-cv-00886 (E.D. Tex. Dec. 9, 2025), ECF No. 93
1016	Int’l Pub. No. WO 2007/021974 A2
1017	U.S. Patent No. 6,301,617 B1
1018	U.S. Patent App. Pub. 2004/0093419
1019	U.S. Patent No. 7,133,905 B2

Exhibit	Reference
1020	U.S. Patent App. Pub. 2010/0281063 A1
1021	U.S. Patent No. 6,330,561 B1
1022	Complaint, *Sandpiper CDN, LLC v. Microsoft Corporation,* No. 2:25-cv-00664-JRG-RSP (E.D. Tex. June 26, 2025), ECF No. 1

*All emphasis is added unless otherwise indicated.

LISTING OF CHALLENGED CLAIMS

Reference	Limitation
Claim 1	
1[pre]	A method of content delivery in a content delivery network comprising at least a first tier of servers, the method comprising:
1[a]	at a first server in the first tier of servers, obtaining a request from a client for a resource, wherein the resource is available as part of a content provider's library;
1[b]	if the resource is not available at the first server or at a peer of the first server, determining if the resource is popular;
1[c]	if the resource is determined to be popular, then the first server obtaining the resource and the first server serving the resource to the client, otherwise,
1[d][i]	if the resource is determined not to be popular, directing the client to a second server in a second tier of servers distinct from the first tier of servers,
1[d][ii]	wherein the second server comprises a first portion of the content provider's library, the first portion comprising at least the resource, wherein at least one other server in the second tier comprises a second portion of the content provider's library, wherein the first portion of the content provider's library is distinct from the second portion, and
1[d][iii]	wherein the second tier is any intermediate tier between the first tier and an origin server that stores resources associated with the content provider's library, and
1[d][iv]	the second server serving the resource to the client.
Claim 2	
2.	A method as recited in claim 1, wherein portions of the content provider's library are logically partitioned across servers in the second tier.

Reference	Limitation
Claim 3	
3.	A method as recited in claim 1, wherein the first portion and the second portion are alphabetically partitioned across servers in the second tier based on a naming convention associated with each distinct resource in the content provider's library.
Claim 4	
4.	A method as recited in claim 1, wherein the content provider's library comprises a plurality of content-types, each resource in the library being associated with one of the plurality of content-types.
Claim 5	
5.	A method as recited in claim 4 further comprising storing distinct portions of the content provider's library on two or more servers in the second tier, wherein the content-types on each of the two or more servers are proportionally distributed, the content-types comprising at least one of audio content, video content, gaming content, and software content.
Claim 6	
6.	A method as recited in claim 1, wherein resources of the content provider's library are distributed in a substantially equal manner across servers in the second tier based on hash values of uniform resource locators (URLs) associated with two or more respective resources in the content provider's library.
Claim 7	
7.	A method as recited in claim 1, wherein the step of directing the client to a second server in a second tier is performed in response to a step of using a hash to determine which second server in the second tier stores the first portion of the content provider's library.
Claim 8	
8.	A method as recited in claim 1 wherein the step of determining whether the resource is popular comprises determining whether a current popularity value for the resource exceeds a popularity threshold.

Reference	Limitation
Claim 9	
9.	A method as recited in claim 8, wherein the first tier of servers comprises a first popularity threshold and the second tier of servers comprises a second popularity threshold, and wherein the first popularity threshold is different from the second popularity threshold.
Claim 10	
10[pre]	A content delivery framework comprising:
10[a]	a plurality of edge servers forming a first tier of servers;
10[b]	a plurality of parent servers forming a second tier of servers, the parent servers being distinct from the edge servers; and
10[c]	at least one popularity service constructed and adapted to obtain a first set of information from at least some of the edge servers about requests for content, and to provide the edge servers with a second set of information about which tier of servers should handle those requests for content, the popularity service making its determinations based, at least in part, on a measure of popularity of requested content; and
10[d]	wherein the requested content is available as part of a content provider's library, and wherein distinct portions of the content provider's library are logically partitioned across parent servers in the second tier.
Claim 11	
11.	A content delivery framework as recited in claim 10, wherein at least one of the parent servers in the second tier comprises a first logically partitioned portion of the content provider's library, the first logically partitioned portion comprising at least the requested content.

Reference	Limitation
Claim 12	
12.	A content delivery framework as recited in claim 11, wherein at least one other parent server in the second tier comprises a second logically partitioned portion of the content provider's library, the second logically partitioned portion being distinct from the first logically partitioned portion.
Claim 13	
13.	A content delivery framework as recited in claim 10, wherein the second set of information further includes identification of a parent server in the second tier that stores a logically partitioned portion of the content provider's library comprising the requested content.
Claim 14	
14.	A content delivery framework as recited in claim 10, wherein the second tier of parent servers is any intermediate tier between the first tier and an origin server that stores the content provider's library.
Claim 15	
15[pre]	A method of content delivery in a content delivery network comprising a plurality of tiers of servers, including at least a first tier of servers and a second tier of servers, the method comprising:
15[a]	at a first server in one tier of the tiers of servers, obtaining a request from a client for a resource, wherein the resource is available as part of a content provider's library;
15[b]	if the resource is not available at the first server or at a peer of the first server, then selectively redirecting the request from the client to a second tier server, the second tier server being in a tier distinct from the tier containing the first server, the redirecting being based, at least in part, on a current popularity value for the resource;

Reference	Limitation
15[c]	wherein the second tier server comprises a first portion of the content provider's library, the first portion comprising at least the resource, wherein at least one other second tier server comprises a second portion of the content provider's library distinct from the first portion, and wherein the second tier of servers is any intermediate tier between the first tier and an origin server that stores the content provider's library, and
15[d]	wherein the step of selectively redirecting the request from the client to a second tier server comprises determining to which second tier server the request should be directed, the determining step using a hash function to identify a second tier server that stores the first portion of the content provider's library.
Claim 16	
16.	A method as recited in claim 15 wherein the first tier of servers comprises a plurality of edge servers and the second tier of servers comprises a plurality of parent servers, and wherein the first server is a parent server in the second tier and the second server is an edge server in the first tier.
Claim 17	
17.	A method as recited in claim 15 wherein the first tier of servers comprises a plurality of edge servers and the second tier of servers comprises a plurality of parent servers, and wherein the first server is an edge server in the first tier and the second server is a parent server in the second tier.
Claim 18	
18.	A method as recited in claim 15 wherein the first tier of servers comprises a plurality of edge servers and the second tier of servers comprises a plurality of parent servers, and wherein the plurality of tiers of servers further comprises an origin tier, and wherein the first server is a parent server in the second tier and the second server is a server in the origin tier.

By 2008, content-delivery networks ("CDNs") gave content owners a way to serve their websites, music, movies, and more to end users over the Internet quickly while avoiding overloading their own servers. CDNs were networks of geographically distributed servers, many of which are provided at the "edge" of the network closest to end users, and others deeper in the network for support. CDNs would store and serve content on behalf of content owners, rather than burdening the owners' servers that may be located far away or could be overwhelmed handling all users' requests.

U.S. Patent No. 9,762,692 (the "'692 patent") (Ex-1001) is directed to a CDN, but it did not offer anything new. Instead, it proposed two plainly intuitive features: (1) store "popular" content on edge servers for the fastest delivery, while leaving "not popular" content deeper in the network; and (2) distribute distinct portions of the content owner's library across those deeper servers, rather than have each server store a complete copy. The Board found during examination that the claims' use of conditionals did not require the latter feature, and the prior art cited in prosecution taught the first. Yet despite a pending anticipation rejection provided by *the Board* as to the first feature, the Examiner allowed the claims based primarily on amendments to the second feature the Board found to be optional. Even so, the second feature was also well-known.

For these reasons, Microsoft Corp. ("Petitioner") petitions for inter partes review ("IPR") of claims 1, 2, 4-15, 17, and 18 ("Challenged Claims") of the '692 patent.

I. GROUNDS FOR STANDING

Petitioner certifies the '692 patent is available for IPR, and Petitioner is not barred or estopped from requesting IPR on the grounds herein. Petitioner files this Petition within one year of service of Patent Owner Sandpiper CDN, LLC ("PO") district court complaint. Ex-1022.

II. IDENTIFICATION OF CHALLENGE

A. Prior Art

Petitioner's grounds rely upon the prior art below based on an assumed priority date of April 4, 2008.[1] The references are analogous to the '692 patent and each other for being in the same field of endeavor (e.g., network servers and/or CDNs) or reasonably pertinent to the problems faced by the inventor (e.g., storing popular content on edge servers and unpopular content on deeper servers).

Prior Art
Seed (U.S. Pub. No. 2006/0015574); published Jan. 19, 2006; prior art under pre-AIA §102(b) (Ex-1005)
Swildens (U.S. Pub. No. 2004/0221019); published Nov. 4, 2004; prior art under pre-AIA §102(b) (Ex-1006)

[1] Petitioner does not concede entitlement to this priority date.

B. Grounds for Challenge

Ground (§103)	Claims	Prior Art
1	1, 2, 4-15, 17, 18	Seed, Swildens
2	1, 2, 4-8, 10-15, 17, 18	Swildens, Seed

III. THE ’692 PATENT

A. Technical Background

To provide end users access to content like websites, music, or movies over the Internet, content owners typically maintain a server that stores their content and transmits it to the end users upon request.

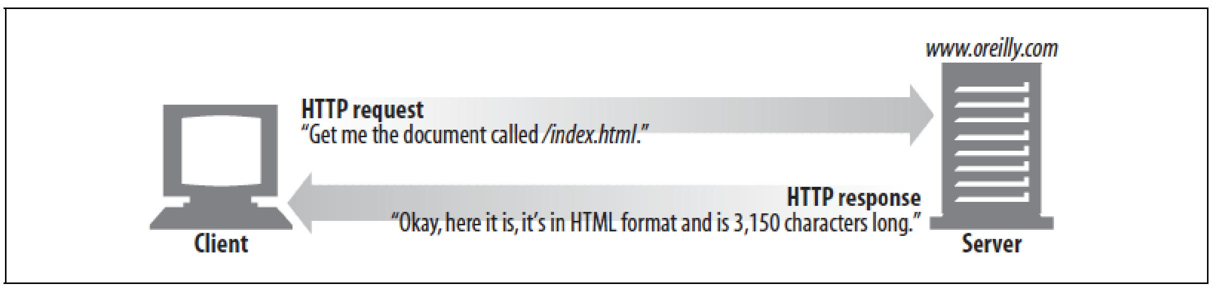


Figure 1-1. Web clients and servers

Ex-1007, Fig. 1-1[2]

Ex-1003, ¶¶35-45; Ex-1007, 4. The content owner’s server is typically called an “origin” server as the originating source of any content. *Id.*, 11; Ex-1003, ¶50.

However, as internet usage grew, “serving Web content from a single location…present[ed] serious problems for site scalability, reliability, and performance,” as servers would overload and crash if they received too many

[2] Images annotated unless otherwise noted.

requests at one time. Ex-1008, 50; Ex-1003, ¶46. Moreover, transmitting content across vast distances often took a significant amount of time. Ex-1003, ¶47. To solve these problems, companies built CDNs that provided additional layers of geographically distributed servers between the end user's device (or "client") and the content owner's origin server. *Id.*, ¶¶48-49; Ex-1009, 9, 16; Ex-1010, Abstract, Figs. 1, 4, ¶¶5-9; Ex-1011, Abstract, 1:15-2:3, Fig. 1. The CDN's servers would "cache" (or store) copies of at least some content and respond to requests on behalf of the content owner, rather than send all requests to the origin server to fulfill. Ex-1003, ¶50; Ex-1009, 5, 37.

To distribute the load and reduce delivery times, CDNs were often arranged in a hierarchy. Ex-1003, ¶¶50-53; Ex-1013, 57; Ex-1009, 9.

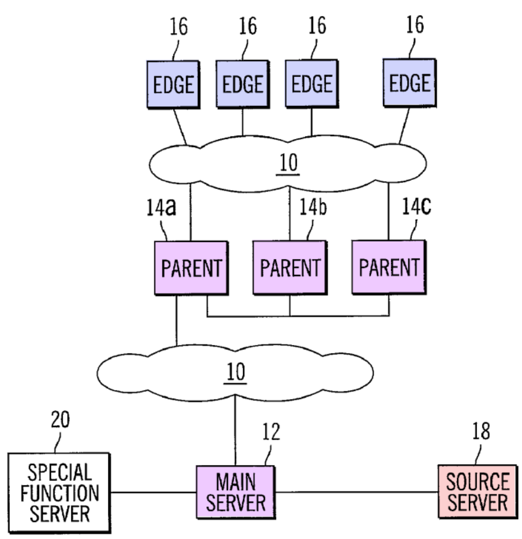


FIG. 2

Ex-1012, Fig. 2. A content owner's origin server (red) would sit furthest from the end user in the hierarchy, and would connect to the CDN. Ex-1003, ¶51. The CDN may then include, e.g., "surrogate" origin servers and/or parent servers (purple) that may be geographically distributed and would cache at least some parts of the content owner's library. *Id*. These surrogate-origin servers and/or parent servers would then feed—either directly or indirectly—clusters of "edge" servers located closest to the end user, which would also cache at least some of the content owner's library. *Id*. The exact number of layers, interactions between different layers or servers within the same layer, and geographic distribution were all well-known design choices based on considerations like speed and cost. *Id.*, ¶¶52-53; Ex-1009, 36-59, 63-64, 67-69; Ex-1011, 3:5-9, 4:9-35, Fig. 3.

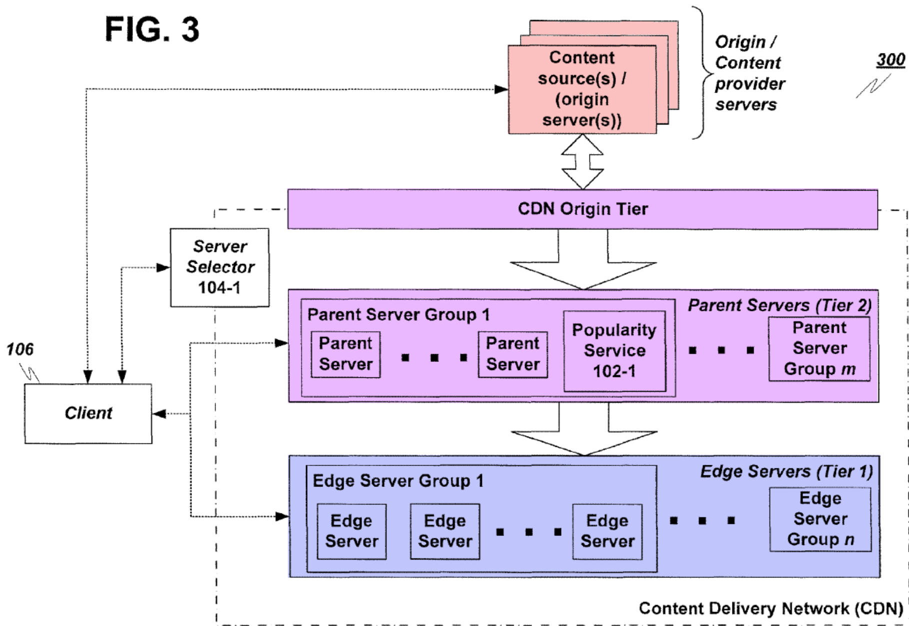
When an end user requests content, an edge server closest to the user would try to fill the request itself, minimizing the delivery time and avoiding burdening the content owner's origin server. Ex-1003, ¶54; Ex-1009, 6; Ex-1008, 50. But if the edge server could not locate the content in its own cache, it could either search for the content on, or redirect the request to, another server (such as another edge, parent, or surrogate origin server) that may store the requested content instead. Ex-1003, ¶¶54-55; 1013, 57. Only if the resource still could not be located would the website owner's origin server be burdened to service the request. Ex-1003, ¶56.

B. Overview

The '692 patent is directed to a CDN with different "tiers" of servers between the end-user (or client) and content owner's origin server (red). Ex-1001, 5:13-29,

Fig. 3; Ex-1003, ¶¶57-61. For instance, the CDN may include clusters of edge servers (blue) connected to parent servers (purple) as well as a CDN-provided surrogate-origin servers (also purple). Ex-1001, 5:13-29, Fig. 3; Ex-1003, ¶61.

FIG. 3



Ex-1001, Fig. 3

The ’692 patent proposed that edge servers would cache “popular,” highly demanded content, while “not popular” content would be stored on servers deeper in the network (e.g., parent or surrogate-origin servers). Ex-1001, Abstract; Ex-1003, ¶62. Moreover, the ’692 patent proposed that not all servers deeper in the network (e.g., parent or surrogate origin servers) need to each store complete copies of a content owner’s library, but can each store distinct portions to reduce the memory required of each server. Ex-1001, 12:17-56; Ex-1003, ¶63.

When an edge server receives a request for content, it would first try to fulfill the request. Ex-1001, Abstract, 5:64-6:3, 11:55-61, Fig. 7; Ex-1003, ¶64. If the edge server could not find the requested content in its cache, it would evaluate whether the content is "popular." Ex-1001, Abstract, 6:13-20, 11:62-64, Fig. 7; Ex-1003, ¶64. If the content is popular, then the edge server *should* be storing a copy in its cache, so it searches for another server that has the requested content, obtains a copy for itself, and also delivers a copy to the end user. Ex-1001, Abstract, 11:65-12:2, Fig. 7; Ex-1003, ¶64. But if the content is not popular, the edge server need not store a copy, so it redirects the end user to another server deeper in the network (e.g., a parent or origin server) to fulfill the request. Ex-1001, Abstract, 12:3-16; Ex-1003, ¶¶65-67.

C. Prosecution History

The Examiner rejected the Challenged Claims based primarily on Seed (U.S. Pub. No. 2006/0015574) and/or Outten (U.S. Pat. No. 7,024,466). Ex-1002, 245-55; Ex-1003, ¶¶68-71. The Examiner cited Seed for teaching a CDN that stored popular content on edge servers and unpopular content on parent servers, and Outten for storing distinct portions of a library across servers deeper in the network. Ex-1002, 245-55; Ex-1003, ¶71. The Applicant argued that Outten did not teach storing *distinct* portions of a library across different servers, (Ex-1002, 295-98), but the Examiner maintained the rejection (*id.*, 2112-27), which the Applicant appealed, (*id.*, 2156; Ex-1003, ¶¶72-74).

On appeal, the Board noted that, in claim 1, "the 'obtaining' step [(1[c])] and the 'directing' step [(1[d][i]-[iv])] are mutually exclusive…the remaining limitations related to the configuration of second tier servers only need to be reached *if* the 'directing' step is reached," and thus the claim "encompasses at least two methods. Ex-1002, 2330-31; Ex-1003, ¶76. For claim 17 (issued independent claim 15), the Board similarly noted that although the "obtaining" step (15[a]) is always performed, the "redirecting" step and "remaining limitations" (15[b]-[c]) "only need to be reached *if* the 'redirecting' step is reached" and thus also encompasses "at least two methods." Ex-1002, 2331-32; Ex-1003, ¶76.

When assessing the prior art, the Board agreed with the Applicant that Outten did not teach storing *distinct* portions of a content library across different servers. Ex-1002, 2333-34; Ex-1003, ¶77. But the Board affirmed the Examiner's rejection in part because: (1) claim 1 did not require "*distinct*" portions be stored across servers, and Outten taught storing portions generally; and (2) Seed alone anticipated claim 17's "obtaining" step without need to reach the "redirecting" step. Ex-1002, 2329-32, 2336-39 (claim 1), 2344-45 (then-pending claim 17); Ex-1003, ¶78.[3]

[3] The Board found that independent claim 1 included a similar conditional, but did not enter an anticipation rejection based on Seed—as it could have—because it instead affirmed the Examiner's combination. Ex-1002, 2329-32.

On remand, however, the Examiner proposed an amendment that merely added “distinct” to limitation 1[d][ii] and added limitation 15[d] directed only to the “redirecting” step, neither of which addressed the Board’s use of conditionals, or even the Board’s anticipation finding of 15[a]. Ex-1002, 2363-68; Ex-1003, ¶79. The Examiner also overlooked that Seed itself disclosed storing distinct portions of a content owner’s library across different servers deeper in the network, and otherwise failed to apply any readily available art that taught this feature as well. This Petition shows the Examiner incorrectly allowed the claims after remand.

IV. LEVEL OF ORDINARY SKILL

A POSITA at the time of the invention would have had at least a bachelor’s degree in computer science, electrical engineering, or a related field, and at least two years of work or research experience in the field of content-delivery management or networks. Work experience can substitute for formal education and vice versa.[4] Ex-1003, ¶¶85-87.

[4] In separate proceedings, PO defined a POSITA to also include “computer engineering” education, and “2-3 years of work experience with content delivery, networking, or the equivalent.” Ex-1014, 6-7. Regardless of which definition is used, the outcome would be the same.

V. CLAIM CONSTRUCTION

No express constructions are necessary; Petitioners have applied the claim terms' ordinary meanings as understood by a POSITA.[5] 37 C.F.R. §42.100(b). However, the Board previously found that independent claims 1 and 15's use of conditionals "encompasses at least two methods"—1[c] versus 1[d][i]-[iv], and 15[a] versus 15[b]-[c] (before 15[d] had been added)—only one of which must be met. Ex-1002, 2329-32. In a suit between PO and a third party, PO argued all steps of claim 1 need not be performed (consistent with the Board), but the third party argued that all steps must be performed, albeit separately. Ex-1014, 12-14; Ex-1015, 9-14. The Court has not resolved that dispute, but regardless, the prior art taught both conditionals as set forth below. Moreover, the third party argued that claiming servers with "distinct portions" of a content library (claims 1/15) is indefinite because the specification clarifies that "distinct portions" may "overlap only minimally," but the outer bounds of "minimally" is unclear. Ex-1015, 7-9. PO argued the plain meaning permits "some overlap." Ex-1014, 9-11. Once again, even if the outer bounds of "distinct portions" is unclear, as set forth below, the prior art suggests *no* overlap, which indisputably constitutes "distinct portions." Finally, the

[5] Petitioners reserve the right to respond to PO/Board constructions. Petitioners reserve §112/claim scope arguments.

third party argued "origin server" (claims 1/15) refers to "a content provider's/customer's server," whereas PO advanced a plain-meaning construction, (Ex-1015, 14), but the prior art taught the third party's specific construction as well.

VI. THE CHALLENGED CLAIMS ARE UNPATENTABLE

A. Ground 1: Seed in view of Swildens (Claims 1, 2, 4-15, 17, 18)

1. Prior Art

a. Seed

Seed discloses a network to deliver website objects using a hierarchy of edge, parent, and origin servers. Seed, [0020], Fig. 2; Ex-1003, ¶80.

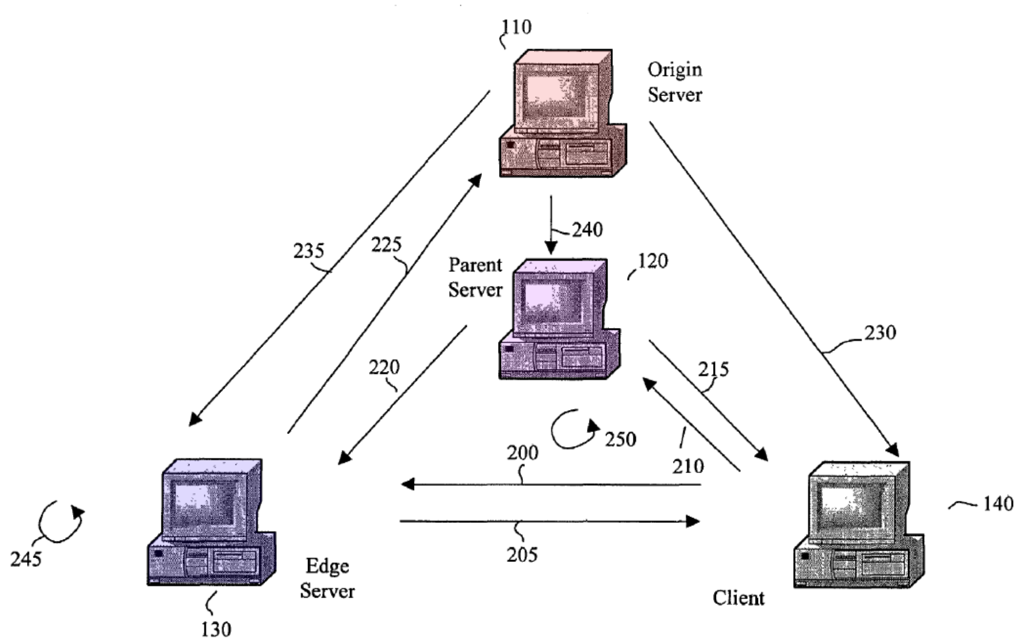


Fig. 2

Seed explains that edge servers are "populate[d]…with the most popular objects," (Seed, [0016]), but "if a requested object is not available at the edge server a parent server that has the requested object will serve the requested object to the clients," (*id.*, [0022]; Ex-1003, ¶81). To carry this out, Seed discloses that "[i]f the selected edge server does not have the requested object, a check is initiated…for the edge server to determine whether the requested object is popular and if so, to replicate the popular requested object to the edge server." Seed, [0018], [0026]; Ex-1003, ¶81. Otherwise, the end user is redirected to a parent server to obtain the object, and the edge server does not replicate it (e.g., because the requested object is not popular). Seed, [0018], [0027]; Ex-1003, ¶81.

Seed also envisions that not every parent will store the same objects, and that parents may use a "recursive[]" redirection technique to search other parents for requested objects. Seed, [0029]; Ex-1003, ¶82.

b. Swildens

Swildens also discloses a network to deliver website files using a hierarchy of edge, origin, and staging servers. Swildens, [0067]-[0070], Fig. 2; Ex-1003, ¶83.

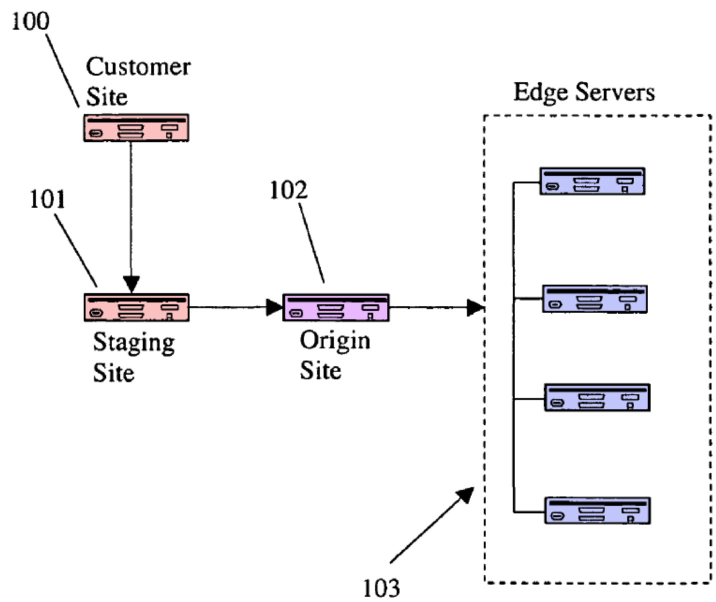


Fig. 1

Swildens discloses that "[a]n edge server maintains a dynamic number of popular files" and the origin server maintains "unpopular" files. Swildens, [0055]; Ex-1003, ¶84. If a user requests a popular file, the edge server fulfills the request, but if the user requests an unpopular file, the user is redirected to the origin server to fulfill the request. Swildens, [0068]-[0069]; Ex-1003, 84. Swildens, however, recognizes that "[a]n individual origin server does not have to be able to hold all of the content on its local storage device," and uses a "hash function" to both partition distinct portions of the unpopular files across multiple origin servers, and also locate which origin server is storing the requested file. Swildens, [0060]-[0066]; Ex-1003, ¶84.

2. Claim 1

a. 1[pre]

If limiting, Seed discloses or suggests this limitation.

Seed discloses "a method…for managed object replication and delivery over a network" (Seed, [0017]), and specifically a CDN, (*id.*, [0010]; Ex-1003, ¶96).

Seed discloses the network comprises "parent server[s]," "edge server[s]," and "origin server[s]" arranged in levels—e.g., edge, parent, and origin levels (or *tiers*, consistent with the '692 patent).[6] Seed, [0010], [0020]-[0022]; *see also id.*, [0017]-[0018], [0024]-[0031], [0062] (describing relationships at each level), Fig. 2; Ex-1001, 2:58-62 ("The Tier 1 servers are also referred to as edge servers….The Tier 2 servers…are also referred to as parent servers."); Ex-1003, ¶¶97-98. Seed also notes that each level may contain "one" or a "cluster" of servers. Seed, [0020]; Ex-1003, ¶97. Seed depicts these server "levels" in Figure 2:

[6] Italics refers to claim language.

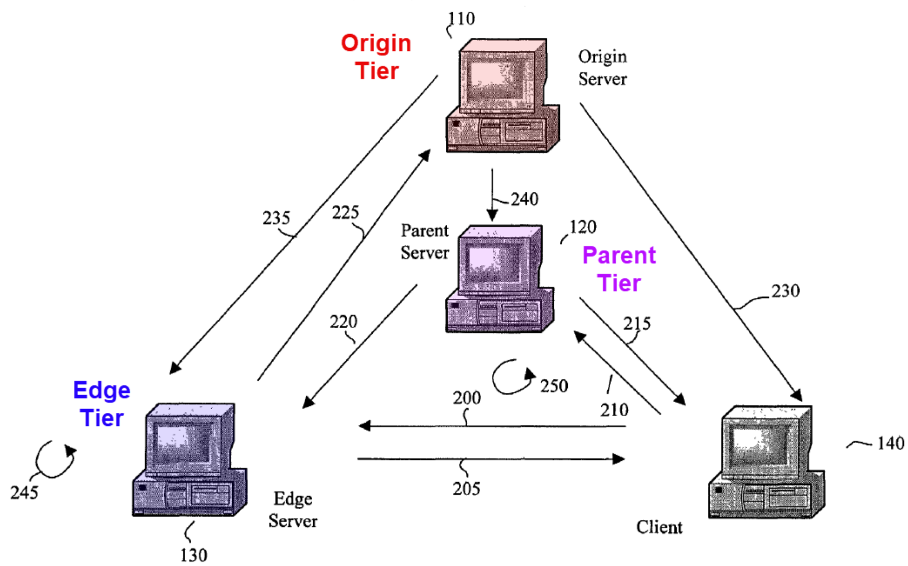


Fig. 2

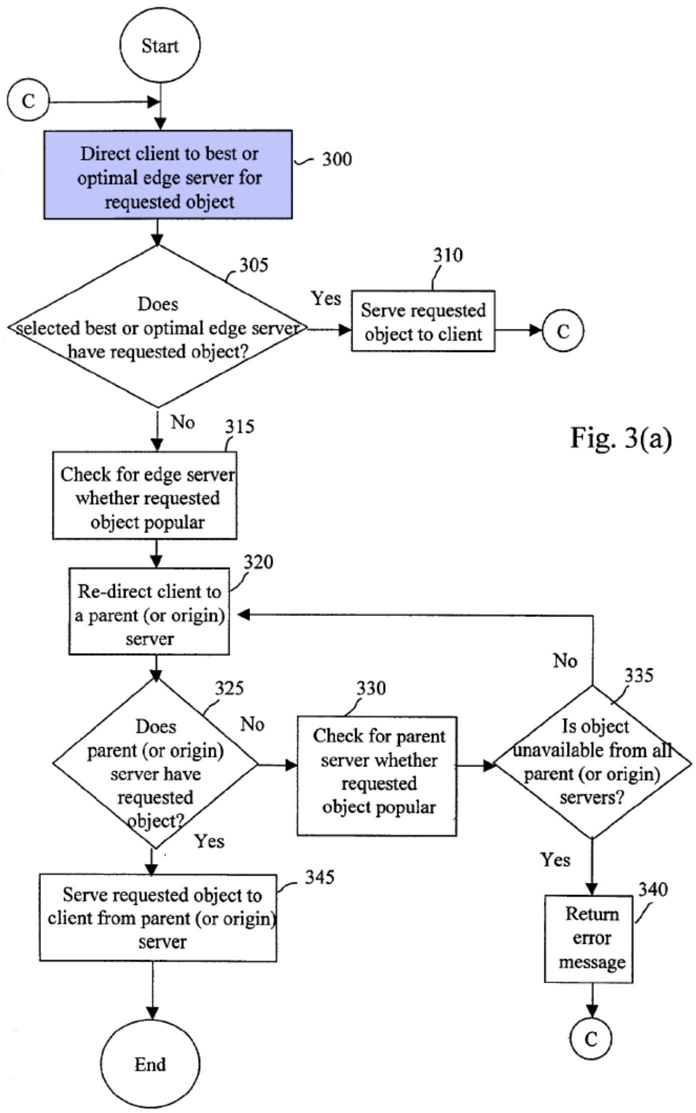
Seed, Fig. 2

Accordingly, Seed discloses method of content delivery in a content-delivery network comprising a level of edge servers (*first tier*). Ex-1003, ¶¶94-98.

b. 1[a]

Seed discloses or suggests this limitation.

Seed discloses that "clients 140 access the network to request one or more objects" (Seed, [0020]), and "a request for a streaming object will be served by a handling edge server if that handling edge server has a copy of that object," (*id.*, [0018]; *see also id.*, [0024] ("the method…directs (at 200, 300) a client, requesting one or more objects, to an edge server in the network"), Fig. 3(a)). In other words, an edge server (*first server*) obtains a request from a client for an object (*resource*). Ex-1003, ¶101.



Seed, Fig. 3(a)

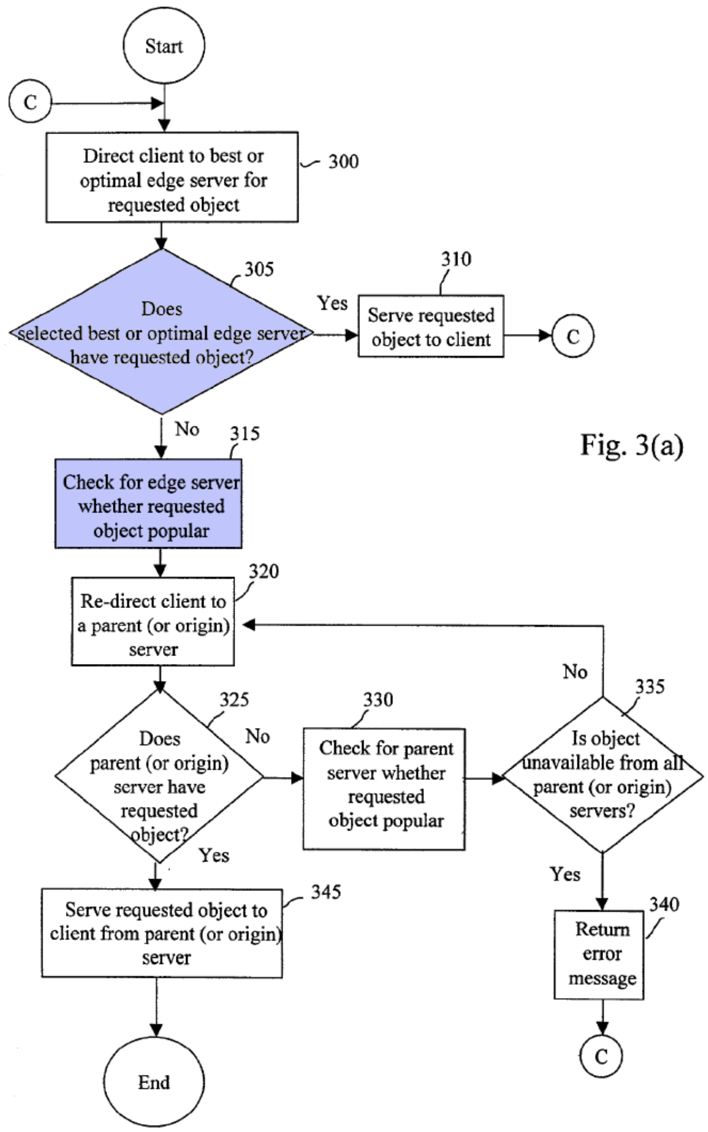
Seed also discloses that "[c]ontent provider companies…that subscribe to the CDN service" maintain an origin server "for storing and serving one or more objects" that are distributed and served through the CDN. *Id.*, [0010], [0020]. A POSITA would have understood that objects (*resources*) are part of a collection of objects (*library*) uploaded by a content provider to the origin server for distribution over the CDN. Ex-1003, ¶102; Ex-1009, 19, 38; Ex-1016, 2:20-3:2, 25:27-26:3, Fig. 2.

Accordingly, Seed discloses an edge server (*first server*) in a level of edge servers (*first tier*), obtaining a request from a client for an object (*resource*) available as part of a content provider's collection of objects (*library*). Ex-1003, ¶¶99-102.

c. 1[b]

Seed discloses or suggests this limitation.

Seed discloses that "[i]f the selected edge server does not have the requested object, a check is initiated (at 315) for the edge server to determine whether the requested object is popular." Seed, [0026], Fig. 3(a); Ex-1003, ¶¶103-104.



Seed, Fig. 3(a)

d. 1[c]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed discloses or suggests this limitation.

Seed discloses that if "the requested object is popular…replicate the popular requested object to the edge server" (*if the resource is determined to be popular, then the first server obtaining the resource*). Seed, [0026]; Ex-1003, ¶107.

Seed discloses that, in response to a first request for a popular but not-yet-replicated object, the client is redirected to another server that has the object, and the object is then replicated to the edge server for "future" requests for the same object. Seed, [0017]-[0018]. However, a POSITA would have understood that clients often request delivery of the same object multiple times (e.g., by revisiting the same website or requesting the same content again). Ex-1003, ¶¶108-109. Thus, for any subsequent client-requests for the object after replication, the edge server would serve the object to the client rather than redirect to another server (*the first server serving the resource to the client*). Ex-1003, ¶109; *Smith & Nephew, Inc. v. Ethicon, Inc.*, 276 F.3d 1304, 1311 (Fed. Cir. 2001) (noting methods "comprising" specified steps "may be practiced with steps in addition to those listed in the claims").

If, however, the claim requires the edge server serve the object to the client after only a first request, Seed also discloses that "checking…whether the requested object is popular and replicating the popular requested object to the edge server may be performed independently of one or more functions of the method of managed object replication and delivery," (Seed, [0027]), suggesting Seed is not limited to redirecting the initial request and separately replicating the popular object, (Ex-1003, ¶¶110-111).

Seed also recognizes that CDN networks routinely configure edge servers to "fetch objects" from other servers "when a requested object is not available on a handling edge server" to serve the object from the edge server that initially received the request. Seed, [0013]; Ex-1003, ¶112; Ex-1009, 25. Although Seed prefers redirecting a first request for popular objects to a different server rather than fetching and serving the popular object itself—to avoid "having [the object] go through the parent path" that "can be slow," (Seed, [0013]-[0014])—Seed nonetheless describes its network as being configured to fetch objects, (*see, e.g.*, *id.*, [0046] (noting "the cost of fetching" is considered as part of the replication policies), [0059] (suggesting partitioning objects into smaller chunks so that "a first part of an object can be served from edge servers quickly, even if most of the object stream ***must be fetched*** from a parent server or origin server."); Ex-1003, ¶112). Based on these disclosures, a POSITA would have understood that Seed suggests having edge servers fetch and then serve popular objects to the client in response to a request, rather than solely redirecting (*the first server serving the resource to the client*). Ex-1003, ¶113.

But even if Seed could not be understood to fetch and serve content in response to a first request, a POSITA would have been motivated to modify Seed to do so. Ex-1003, ¶114. A POSITA would have understood that there were only three typical ways for a server to ensure that requests for an object missing from its cache are fulfilled: (1) obtain (fetch) the requested object from another server and then serve it to the client; (2) redirect the client to another server that has the object; or (3) forward the request to another server that has the object. Ex-1003, ¶¶114-115; *see, e.g.*, Ex-1017, 2:38-42 ("the server…locate[s] the best current location for the resource and either redirects the requesting browser to it…or else fetches and returns the resource"); Ex-1018, [0040] ("edge server 508 can forward requests to the origin, fetch content from the origin to deliver to the end user, and/or deliver content from its own cache as required."). Because there were only a finite number of ways to fulfill requests for missing content, a POSITA would have been motivated to try fetching-and-serving as one of them. Ex-1003, ¶115; *Perfect Web Techs., Inc. v. InfoUSA, Inc.*, 587 F.3d 1324, 1331 (Fed. Cir. 2009) (finding obvious to try three known, predictable ways to meet an e-mail quota). Alternatively, replacing Seed's redirecting with fetching would have merely amounted to a simple substitution of one known element (redirecting-and-fetching) for another (fetching-and-serving), or replacing one known technique (redirecting-and-fetching) with another suitable option (fetching-and-serving) to achieve a predictable result or solve a known problem in the art (serving requested but missing objects). Ex-1003, ¶115; *Intel Corp. v. PACT XPP Schweiz AG*, 61 F.4th 1373, 1380 (Fed. Cir. 2023); *Snap Inc. v. You Map, Inc.*, No. 24-112, 2025 WL 2793538, *3-4 (Fed. Cir. Oct. 1, 2025).

Despite Seed's expressed preference for redirecting, a POSITA would have also been specifically motivated to replace Seed's redirect-and-fetch process with a fetch-and-serve process because fetching provides different benefits. Ex-1003, ¶116; *In re Gurley*, 27 F.3d 551, 553 (Fed. Cir. 1994) ("A known or obvious composition does not become patentable simply because it has been described as somewhat inferior to some other product for the same use."); *Gen. Elec. Co. v. Raytheon Techs. Corp.*, 983 F.3d 1334, 1352 (Fed. Cir. 2020) (prior-art preference for one option does not render another known option non-obvious). For instance, fetching would reduce CDN-processing and transmission costs by avoiding having to transfer the same object twice from another server—once to the client, and again to the edge server replicating the object later. Ex-1003, ¶116; Swildens, [0043] ("The customer is charged for the bandwidth used to transfer files from the origin servers to the edge servers").

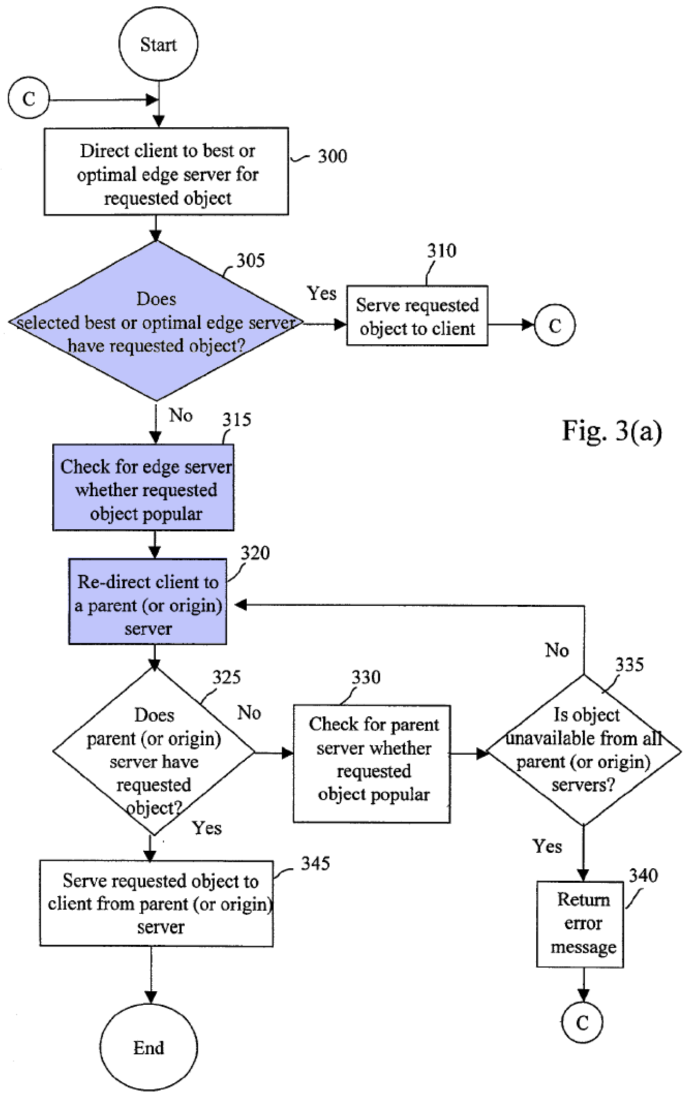
A POSITA would have had a reasonable expectation of success in making this modification because both fetching and redirecting were well-known, and swapping one for another merely involves changing which network system obtains the requested object; rather than have an edge server tell the client where to look (redirecting), the edge server fetches the object itself from that location (which Seed's edge server is already programmed to do, only ***after*** redirecting) and then provide it to the client. Ex-1003, ¶117; Seed, [0046], [0059]; Ex-1017, 2:38-42. The minimal changes to computer code needed to make this modification would have been well within the skills of a POSITA to implement. Ex-1003, ¶117.

Accordingly, Seed discloses or renders obvious that if the object (*resource*) is popular, the edge server (*first server*) replicates (*obtains*) the object and serves the object to the client either in response to a subsequent request or as part of a fetch-and-serve process. Ex-1003, ¶¶105-117.

e. 1[d][i]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed discloses or suggests this limitation.

Seed discloses that edge servers are populated "with the most popular objects" and "the rest" are served from "parent servers." Seed, [0016]. Seed discloses that when the edge server does not have the requested object—because the object either is not popular, or the edge server has not yet determined that the object is popular, Ex-1003, ¶120—"the selected edge server directs (at 210, 320) the requesting client 140 to a parent server 120" among parent-level servers. Seed, [0028], Fig. 3(a); §VI.A.2.a (1[pre]).



Seed, Fig. 3(a)

Accordingly, Seed discloses if the resource is determined not to be popular, directing the client to a parent server (*second server*) among parent-level servers (*second tier*) distinct from the edge-level servers (*first tier*). Ex-1003, ¶¶118-120.

f. 1[d][ii]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed alone or combined with Swildens teaches or suggests this limitation.

Seed discloses that when the edge server redirects the client to a parent server, (§VI.A.2.e (1[d][i])), "if a parent server does not have the requested object, the parent server could itself use a redirection technique recursively…until a final parent server is reached that has the requested object." Seed, [0029]. In other words, Seed recognizes that not every parent-level server has the same objects from the content-provider's collection (*library*), and some parent servers (*second server*) may have portions with the requested object (*a first portion of the content provider's library, the first portion comprising at least the resource*), and other parent servers (*one other server in the second tier*) have distinct portions—even if some overlap, consistent with PO's view of the claim (§V (Claim Construction))—that do not include the requested object (*wherein the first portion of the content provider's library is distinct from the second portion*). Seed, [0029]; Ex-1003, ¶¶123-124.

But even if Seed does not expressly disclose this limitation, it would have been obvious in view of Swildens. Ex-1003, ¶125. Swildens teaches another CDN in which "edge server[s] maintain[] a dynamic number of popular files," while its intermediate (what Swildens calls "origin") servers store unpopular files.[7] Swildens,

[7] Whereas Seed discloses "edge," "parent," and "origin" tiers, Swildens teaches "edge," "origin," "staging," and "customer" tiers. *See, e.g.*, Swildens, Fig. 1. A POSITA would have understood that because Swilden's "origin" tier is an

[0043], [0055]; Ex-1003, ¶¶125-130. Swildens, however, recognizes that although intermediate "servers together store the complete copies of all the content for a particular customer," (*id.*, [0050]), "[a]n individual [intermediate] server does not have to be able to hold all of the content," (*id.*, [0060]). Instead, Swildens uses a "hash function"—an algorithm to "map[] the large number of content identifiers across the much smaller number of storage servers," (*id.*, [0061])—to distribute distinct portions of a content provider's library across different intermediate servers and identify on which server any given content resides, (*id.*, [0060]; Ex-1003, ¶¶131-134). Unless configured specifically to allow overlap for redundancy (which Swildens recommends for improving performance, but does not require, Swildens, [0058]), a POSITA would have understood that a hash function would typically result in each server having distinct, non-overlapping portions of the library, because each content identifier would be assigned to one server based on its hash value. §V (Claim Construction); Ex-1003, ¶134.

A POSITA would have been motivated to implement a hash function like Swildens' to distribute distinct portions of a content provider's library across Seed's

---

intermediate-level server intended to store unpopular content, its teachings would obviously apply to Seed's intermediate-level "parent" tier of servers also intended to store unpopular content. Ex-1003, ¶¶126-130, 138.

parent-level servers to reduce parent-server-storage costs. Ex-1003, ¶135. Specifically, Swildens explains that "[t]he customer is charged for the amount of storage required on the [intermediate] site 102 to store the customer's files" as well as "the bandwidth used to load the files onto the [intermediate] site 102 and edge servers 103." Swildens, [0056]. Providing each intermediate server with only a (distinct) portion of a content library—and particularly unpopular content—reduces how much storage is required at each intermediate server (thereby reducing storage and upload costs). *Id.*, [0058], [0095]-[0101]; Ex-1003, ¶135.

Furthermore, incorporating Swildens' hash function into Seed's CDN provides a more-efficient way of locating unpopular content distributed across the intermediate-tier servers. Ex-1003, ¶136. Rather than repeated, recursive redirecting to locate an object across several intermediate servers (Seed, [0029]), Swildens' hashes are incorporated into the uniform-resource locator ("URL") used to redirect the client to the specific intermediate server that is expected to have the object in the first instance (Swildens, [0137]; Ex-1003, ¶136). A POSITA would have appreciated that incorporating a hash function like Swildens' into Seed would allow clients to be redirected to the appropriate intermediate (parent) server faster, leading to faster service of unpopular objects. Ex-1003, ¶136.

A POSITA would have had a reasonable expectation of success in combining Seed and Swildens because Seed already envisions distributing portions of content libraries across an intermediate (parent) tier of servers; Swildens simply addresses the benefits of purposefully distributing distinct portions, and provides an efficient means of both distributing those distinct portions and then locating requested content using hash functions. Ex-1003, ¶¶137-138. In effect, the modification would require changing how Seed assigns URLs to both distribute and then later locate content in the parent tier of servers by incorporating a hash into those URLs. Ex-1003, ¶137. Such a modification would have been easily implementable to a POSITA. Ex-1003, ¶137; Ex-1009, 58.

Accordingly, Seed, alone or combined with Swildens, renders obvious a parent server (*second server*) that stores one (*first*) portion of the content provider's library that has the requested object (*resource*), and another parent server (*other server*) in the parent-level servers (*second tier*) stores a distinct (*second*) portion of the content provider's library. Ex-1003, ¶¶121-138.

g. 1[d][iii]

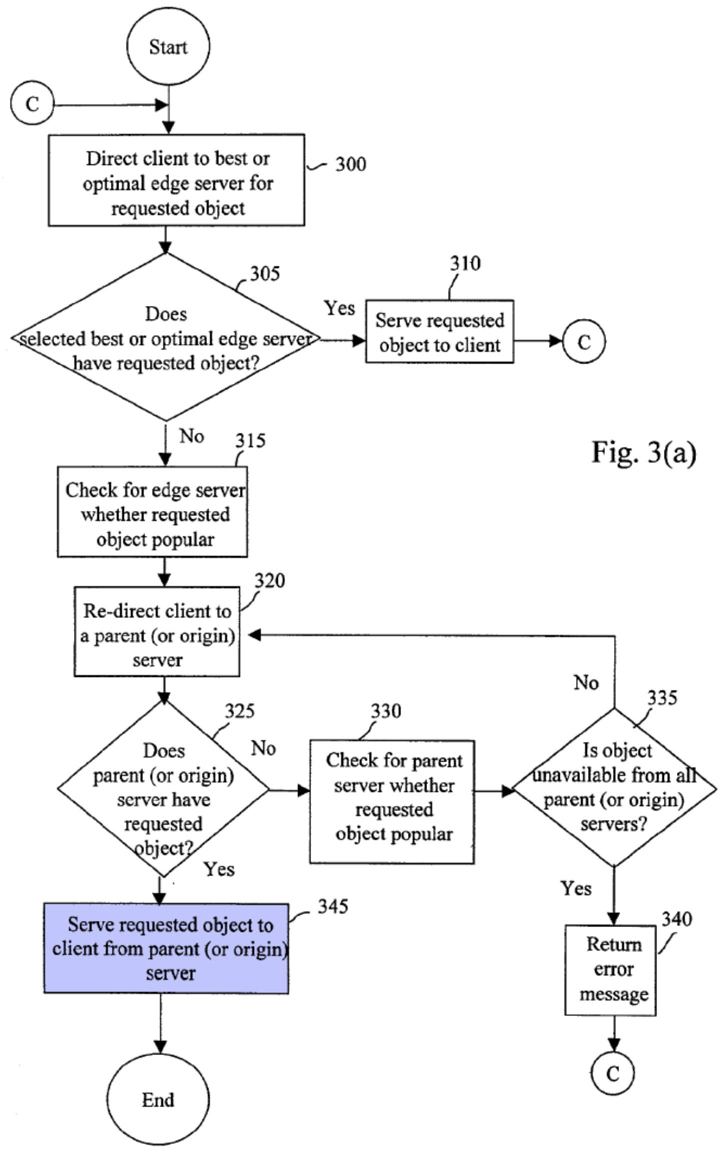
If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed discloses or suggests this limitation.

Seed discloses that parent-level servers (*second tier*) are intermediate to the edge-level servers (*first tier*) and origin server (e.g., customer's server). §VI.A.2.a (1[pre]); Ex-1003, ¶¶139-140. Seed also discloses that "[c]ontent provider companies…that subscribe to the CDN service" maintain an origin server "for storing and serving one or more objects" (*an origin server that stores resources associated with the content provider's library*). Seed, [0010], [0020]; Ex-1003, ¶141.

h. 1[d][iv]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed discloses or suggests this limitation.

Seed discloses the parent server (*second server*) serves an unpopular object (*resource*) to the client. Seed, [0028] ("the client 140 is redirected to a parent server that has the requested object and is able to serve…the requested object to the client"), [0029], Fig. 3(a); Ex-1003, ¶¶142-143.



Seed, Fig. 3(a)

3. Claim 2

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed combined with Swildens teaches or suggests this additional limitation.

Although Seed envisions storing different portions of a content library across parent-level servers (*second tier*) (Seed, [0029]; §VI.A.2.f (1[d][ii]), Seed does not expressly describe how a content library is partitioned across parent servers (Ex-1003, ¶¶144-145).

Swildens, however, teaches partitioning a content library across intermediate-level servers (*second tier*) using a hash function, and specifically "comput[ing] a hash…from a content file identifier" and storing the file at an intermediate-level server (like Seed's parent servers) based on the hash. Swildens, [0051]-[0052]. Swildens also teaches that "[a] good hash function is one that spreads the content storage requirements of the content identifiers across the actual storage capacities of each of [intermediate] server group." *Id.*, [0061]; Ex-1003, ¶146.

In other words, Swilden teaches partitioning and distributing portions of a content provider's library across intermediate-level servers (like Seed's parent level; *second tier*) using a hashing algorithm (*logic*). Ex-1003, ¶147. And for the reasons discussed above, it would have been obvious to implement Swildens' hash function and partitioned libraries in Seed's CDN. §VI.A.2.f (1[d][ii]); Ex-1003, ¶148.

4. Claim 4

Seed discloses or suggests this additional limitation.

Seed discloses "an object includes, without limitation, an audio file…, a video file…, an image file…and any other software or data file or object," with specific example file formats. Seed, [0010].

A POSITA would have understood that each of these are “content types” consistent with the ’692 patent (Ex-1001, 12:29-31), and that each object (*resource*) would be associated with one of the content types, (Ex-1003, ¶¶149-151).

5. Claim 5

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed combined with Swildens teaches or suggests this additional limitation.

Seed, alone or combined with Swildens, teaches storing distinct portions of a content provider’s library across multiple parent-level servers (*second tier*). §§VI.A.2.e-VI.A.2.f (1[d][i]-[ii]); Ex-1003, ¶¶152-153.

Seed also discloses the library may include multiple content types, including audio, image, video, and software. §VI.A.4 (cl. 4); Ex-1003, ¶154.

When combining Seed and Swildens to partition content libraries into distinct portions across multiple parent-level servers, a POSITA would have appreciated that the content types of objects would be distributed on each of Seed’s parent-level servers by some proportion (e.g., some parent servers would have some proportion of objects of a certain content type). Ex-1003, ¶155. Indeed, distributing content across servers by type was known as a means to balance load. *Id.* (citing Ex-1020, [0038] (spreading different video files over different servers); Ex-1021, 8:25-56).

6. Claim 6

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed combined with Swildens teaches or suggests this additional limitation.

Seed combined with Swildens teaches storing distinct portions of a content provider's library across multiple parent-level servers (*second tier*) using a hash function. §§VI.A.2.e-VI.A.2.f (1[d][i]-[ii]); Ex-1003, ¶¶156-157.

As part of the distribution, Swildens explains that "[a] good hash function is one that spreads the content storage requirements of the content identifiers across the actual storage capacities of each [intermediate] server group." Swildens, [0061]. A POSITA would have appreciated that when Seed is implemented in a network with parent servers that have the same amount of storage, Swildens' hash function would result in a substantially equal distribution of content across the parent-level servers (*servers in the second tier*). Ex-1003, ¶158.

Moreover, Swildens further explains that hash values are determined based on "content identifiers" or "published URLs" associated with the content (Swildens, [0061]-[0062], [0065]-[0066]), and "[r]edirected requests to [intermediate] servers have a hash included in the URL" that is used to locate which intermediate server is storing the requested object, (Swildens, [0060]). As such, Seed combined with Swildens teaches distributing content across the parent-level servers (*servers in the second tier*) based on hash values either included in URLs, or of URLS themselves (*based on hash values of uniform resource locators (URLs)*) associated with content via content identifiers or their published URLs (*associated with two or more respective resources in the content provider's library*). Ex-1003, ¶159.

7. Claim 7

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Seed combined with Swildens teaches or suggests this additional limitation.

Although Seed discloses using recursive redirection to direct the client to a parent server in the parent-level servers (*second tier*) that has the requested object, it would have been obvious to use a hash function to determine which parent server (*second server*) is storing the requested object in its portion of the library, as taught by Swildens. §§VI.A.2.e-VI.A.2.f (1[d][i]-[ii]); Ex-1003, ¶¶160-161.

8. Claim 8

Seed discloses or suggests this additional limitation.

Seed discloses "an object is replicated (at 520) into storage"—e.g., that the object is popular—"when the popularity P of the object is greater (at 500) than the initial threshold." Seed, [0044]; *see also id.*, [0035]-[0040] (providing algorithm for calculating popularity), [0041]-[0043] (recommending dynamic rather than fixed thresholds); Ex-1003, ¶¶162-163.

9. Claim 9

Seed discloses or suggests this additional limitation.

Seed discloses using a dynamic threshold that depends on the storage capacity of each server (Seed, [0041]-[0044]), and notes that both edge and parent servers determine thresholds, (*id.*, [0034], [0044], Fig. 3(b) (noting "purge [less-popular] object(s) from … parent server"), Fig. 5 (dynamic threshold determination)); Ex-1003, ¶¶164-165.

Accordingly, a POSITA would have understood that edge servers among edge-level servers would have thresholds based on their own storage capacities (*the first tier of servers comprises a first popularity threshold*), and parent servers among parent-level servers would have other thresholds based on their own storage capacities (*the second tier of servers comprises a second popularity threshold, wherein the first popularity threshold is different from the second*). Ex-1003, ¶166.

10. Claim 10

a. 10[pre]

Seed discloses or suggests this limitation.

Seed discloses or suggests a content-delivery framework, including a CDN and method for using one. *See* §VI.A.2.a (1[pre]); Ex-1003, ¶¶167-168.

b. 10[a]

Seed discloses or suggests this limitation.

Seed discloses or suggests a plurality of edge servers (*first tier*). *See* §VI.A.2.a (1[pre]); Ex-1003, ¶¶169-170.

c. 10[b]

Seed discloses or suggests this limitation.

Seed discloses or suggests a plurality of parent servers (*second tier*). *See* §VI.A.2.a (1[pre]); Ex-1003, ¶¶171-172.

d. 10[c]

Seed combined with Swildens teaches or suggests this limitation.

Seed discloses that each edge and parent server makes its own popularity determinations in response to their individual thresholds. *See, e.g.*, Seed, [0026], [0029], [0041]-[0044]. Seed does not expressly disclose a separate "popularity service" that receives information from edge servers and provides information back to edge servers. Ex-1003, ¶¶173-174.

Swildens, however, teaches that rather than solely having each edge server make its own popularity determination, in an alternative embodiment, "an auto migration server [(*popularity service*)] collects all of the log files…from the edge servers [(*constructed and adapted to obtain a first set of information from…the edge servers*)]…and tabulates the miss count…for every piece of content" (*the popularity service making its determinations based…on a measure of popularity of the requested content*) to determine whether any content is sufficiently popular based on a threshold such that it should be auto-migrated to the edge (*provide the edge servers* *with a second set of information about which tier of servers should handle those requests for content*). Swildens, [0126], [0128], [0130]; Ex-1003, ¶175.

A POSITA would have been motivated to implement Swildens' "auto migration server" in Seed's CDN to determine object popularity based on aggregated (miss count) information from multiple edge servers, rather than solely having each edge server determine popularity individually, to improve network performance for, e.g., content of rapidly increasing popularity. Ex-1003, ¶176.

A POSITA would have had a reasonable expectation of success in making this modification because it merely involves configuring a centralized server—such as Seed's parent server(s), or a separate "auto migration server"—to compile miss-count information from the edge servers that they would each already have to make a centralized popularity determination that would apply to multiple edge servers. Ex-1003, ¶177.

e. 10[d]

Seed discloses or suggests this limitation.

Seed discloses or suggests that the requested content is available as part of a content provider's library (§VI.A.2.b (1[a])), and Seed combined with Swildens teaches or suggests storing distinct portions of the content provider's library that are partitioned using a hash function (*logically*) across parent-level servers (*second tier*). §§VI.A.2.f (1[d][ii]), VI.A.3 (cl. 2); Ex-1003, ¶¶178-179.

11. Claim 11

Seed combined with Swildens teaches or suggests this limitation.

Seed combined with Swildens teaches or suggests storing distinct portions of the content provider's library that are partitioned using a hash function (*logically*), including a first portion with the requested content, on one parent server in the parent-level servers (*second tier*). §§VI.A.2.f (1[d][ii]), VI.A.3 (cl. 2); Ex-1003, ¶¶180-181.

12. Claim 12

Seed combined with Swildens teaches or suggests this limitation.

Seed combined with Swildens teaches or suggests storing distinct portions of the content provider's library that are partitioned using a hash function (*logically*), including a second portion on another parent server among the parent-level servers (*second tier*) distinct from the first portion. §§VI.A.2.f (1[d][ii]), VI.A.3 (cl. 2); Ex-1003, ¶¶182-183.

13. Claim 13

Seed combined with Swildens teaches or suggests this limitation.

Seed combined with Swildens teaches or suggests using a hash function (*logic*) to partition and then identify which among a plurality of parent servers (*second tier*) is storing a portion of a content provider's collection (*library*). §§VI.A.2.f (1[d][ii]), VI.A.3 (cl. 2), VI.A.10.e (10[d]); Ex-1003, ¶185. Seed combined with Swildens also teaches using a central server (like Swildens' auto-migration server) to make collective popularity determinations for a plurality of edge servers. §VI.A.10.d (10[c]); Ex-1003, ¶185.

Upon incorporating a hash function and central server to make collective popularity determinations into Seed's system, a POSITA would have appreciated that the central server would send to edge servers information that maps the content identifiers to the parent server in a parent level of servers that would store the portion of the content provider's collection (*library*) that stores the requested content. *See* Swildens, [0065]; Ex-1003, ¶¶184-186.

14. Claim 14

Seed discloses or suggests this limitation.

Seed discloses or suggests the parent-level servers are intermediate to the edge servers (*first tier*) and an origin server that stores the content provider's library. §VI.A.2.g (1[d][iii]); Ex-1003, ¶¶187-188.

15. Claim 15

a. 15[pre]

If limiting, Seed discloses or suggests this limitation.

Seed discloses a method of content delivery in a CDN comprising edge-level (*first tier*), parent-level (*second tier*), and origin-level servers (collectively, *a plurality of tiers*). §VI.A.2.a (1[pre]); Ex-1003, ¶¶189-190.

b. 15[a]

Seed discloses or suggests this limitation, which may be mapped in at least two ways based on dependent claims 17 and 18. Specifically, dependent claims 17 and 18 both specify the first-tier servers are edge servers and the second-tier servers are parent servers. But dependent claim 17 specifies the first server is an ***edge*** server, and dependent claim 18 specifies the first server is a ***parent*** server. Ex-1003, ¶¶191-192.

For dependent claim 17, Seed discloses an edge server (*first server*) among edge-level servers (*one tier of the tiers of servers*) obtaining a request from a client for an object (*resource*) available as part of a content provider's library. §VI.A.2.b (1[a]); Ex-1003, ¶193.

For dependent claim 18, Seed discloses a parent server (*first server*) among parent-level servers (*one tier of the tiers of servers*) obtaining a request from a client for an object (*resource*), (Seed, [0028] (redirecting request to parent)), available as part of a content provider's library, (§VI.A.2.b (1[a]); Ex-1003, ¶194).

c. 15[b]

If 15[b] must be performed (§V (Claim Construction)), Seed discloses or suggests this limitation, which may be mapped in at least two ways. Ex-1003, ¶¶195-196.

For dependent claim 17, Seed discloses that if the object (*resource*) is not available at the edge server (*first server*), the request is redirected to a parent-level server (*second-tier server*) distinct from the edge-level servers (*first tier*) containing the edge server (*first server*). §§VI.A.2.a (1[pre]), VI.A.2.e (1[d][i]); Ex-1003, ¶197. Seed also discloses the request is redirected when objects do not have a current popularity value that exceeds a popularity threshold. §§VI.A.2.d (1[c]) (storing popular content at edge), VI.A.2.e (1[d][i]) (edge servers redirecting to parent servers for unpopular content), VI.A.8 (cl. 8) (using a popularity value for the popularity determination); Ex-1003, ¶197.

Dependent claim 18, however, appears contradictory when read with this limitation. Claim 18 specifies that "the first server is a parent server in the second tier" and "the second tier of servers comprises a plurality of parent servers." Limitation 15[b] and claim 18 thus collectively require that "if the resource is not available at [a parent server ***in the second tier***]…then selectively redirecting the request from the client ***to a second tier server*** [(e.g., another parent server)], the second tier server ***being in a tier distinct from the tier containing the [the parent server in the second tier]***." At base, these limitations require redirecting a request from one (second-tier) parent server to another (second-tier) parent server, yet in distinct tiers. Ex-1003, ¶198.

For purposes of this petition, these limitations may be jointly interpreted any of three ways:

(1) the request is redirected to a peer-parent server, which would ignore that the two must be in distinct tiers as required by the claim;

(2) the request is redirected to a distinct "second tier" of parent servers (e.g., the parent servers have a multi-tier substructure); or

(3) The request is redirected to "a second server"—not a "second tier server" as claimed—which claim 18 specifies is an origin server, which would comply with the requirement that the first (parent) server and second (origin) server are in distinct tiers.

Ex-1003, ¶¶199-203.

Regardless of which interpretation applies, Seed discloses each. *Intel Corp. v. Qualcomm Inc.*, 21 F.4th 801, 813 (Fed. Cir. 2021) ("[t]he indefiniteness of a limitation...precludes a patentability determination only when the indefiniteness renders it logically impossible for the Board to reach…a decision" on the prior-art grounds).

(1) Seed discloses recursively redirecting a request among multiple parent servers in the second tier (i.e., a parent server (*first server*) among parent-level servers (*second tier*) to at least another parent server also among parent-level servers (*second-tier server*)). Seed, [0029]; Ex-

1003, ¶200. However, when combining Seed with Swildens to use hash functions instead of recursive redirection (§VI.A.2.f (1[d][ii]), a POSITA would have appreciated that one parent server may still use a hash function to redirect to another parent server if the original recipient does not have the requested content. Swildens, [0046] (noting redirection to an intermediate "server that ***should*** have the content"); Ex-1003, ¶201.

(2) Seed also discloses that "[a]s will be apparent to those skilled in the art, more than one layer or level of parent servers…may be used." Seed, [0022]. Seed therefore discloses multiple sub-layers (*tiers*) of parent servers between the edge and origin tiers. Ex-1003, ¶202. When redirecting from one parent server (*first server in the second tier*) to another (*second-tier server*) (Seed, [0029]), it would have been obvious to redirect a request from one level of parent servers (*second tier*) to a distinct level of parent servers (a distinct *second tier*), if, for instance, different sub-layers of parent servers store content of different degrees of popularity. Ex-1003, ¶202. For instance, one layer of parent servers closer to the edge may store objects that are less popular than those at the edge, but more popular than those stored by a lower-tier of parent servers further from the edge. *Id*.

(3) Finally, Seed discloses that "where the parent servers [(*first server in the second tier*)] collectively are not populated with all of the objects and the network has access to the origin server of a requested object, the client may be redirected (at 225, 320) to the origin server ([*second server*)] if the requested object is not available on the parent servers," Seed, [0031], at a distinct origin level (*tier*) from the parent servers (*second tier*), §VI.A.2.a (1[pre]); Ex-1003, ¶203.

Under any possible interpretation, Seed discloses this limitation. Ex-1003, ¶204. Moreover, under any interpretation, redirecting would be *based, at least in part, on a current popularity value for the resource* because Seed discloses that the parent servers redirect requests only *after* the edge server has determined an object is not popular. Seed, [0026]; Ex-1003, ¶204.

d. 15[c]

If 15[c] must be performed (§V (Claim Construction)), Seed alone or combined with Swildens teaches or suggests this limitation.

For both dependent claims 17 and 18, Seed combined with Swildens discloses partitioning the content provider's library across parent servers (*second-tier servers*), with each parent server storing distinct portions of the library, including one parent server with one portion with the requested object (*resource*), and a separate parent server with a distinct portion, with that the parent servers (*second tier*) being intermediate to the edge-level servers (*first tier*) and an origin server (e.g., customer's server) that stores the library. §§VI.A.2.f-VI.A.2.g (1[d][ii]-[iii]); Ex-1003, ¶¶205-207.

e. 15[d]

If 15[d] must be performed (§V (Claim Construction)), Seed combined with Swildens teaches or suggests this limitation.

For both dependent claims 17 and 18, Seed combined with Swildens discloses redirecting a request from the client to a parent server (*second-tier server*) by determining to which parent server the request should be redirected using a hash function to identify the parent server that stores the portion of the content provider's library with the requested object. §§VI.A.2.e-VI.A.2.f (1[d][i]-[ii]; redirecting), VI.A.7 (cl. 7; hash function); Ex-1003, ¶¶208-210.

16. Claim 17

Seed combined with Swildens teaches or suggests the method of claim 15, in which the first-tier servers are edge servers, the second-tier servers are parent servers, the first server is an edge server, and the second server is a parent server. §§VI.A.15.a-VI.A.15.e (15[pre]-[d]); Ex-1003, ¶¶211-212.

17. Claim 18

Seed combined with Swildens teaches or suggests the method of claim 15, in which the first-tier servers are edge servers, the second-tier servers are parent servers, the first server is a parent server, and the second server is either another parent server or an origin server. §§VI.A.15.a-VI.A.15.e (15[pre]-[d]); Ex-1003, ¶¶213-214.

B. Ground 2: Swilden in view of Seed (Claims 1, 2, 4-8, 10-15, 17, 18)

1. Claim 1

a. 1[pre]

If limiting, Swildens discloses or suggests this limitation.

Swildens discloses "[a] method…for the automatic migration of data via a distributed computer network [that] allows a customer to select content files that are to be transferred to a group of edge servers" for delivery to users. Swildens, Abstract, [0018]; *see also id.*, [0006] (providing "for the intelligent distribution of an information provider's multimedia files across the Internet."); Ex-1003, ¶217.

Although Swildens does not expressly use the words "content delivery network," Swildens repeatedly describes processes for delivering content to users through the network. *See, e.g.*, *id.*, [0031]-[0039], [0046], [0135]-[0137], Fig. 5 (depicting steps for "Delivery of popular content"), Fig. 6 (depicting steps for "Delivery of unpopular content"); Ex-1003, ¶218. Swildens also depicts example URLs for delivering content that incorporate the acronym "cdn"—a reference to a "content delivery network" as claimed. *See, e.g.*, Swildens, Fig. 5; Ex-1003, ¶218.

Swildens also discloses the network comprises tiers of "edge servers," "origin servers," a "staging server," and a "customer's server." Swildens, [0042], [0052]-[0055], Figs. 1, 2, 5, 6; Ex-1003, ¶219.

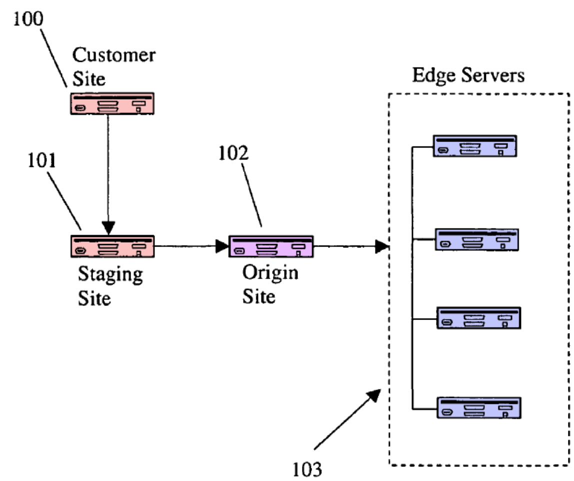
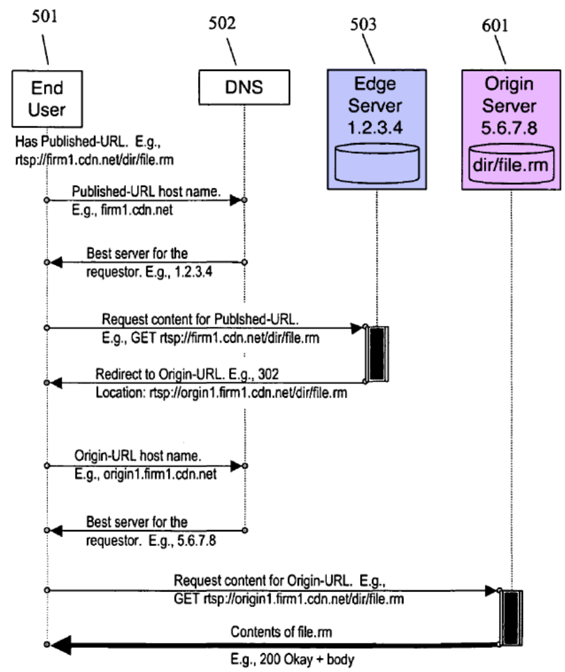


Fig. 1



Delivery of unpopular content
Fig. 6

Swildens, Figs. 1, 6

Accordingly, Swildens discloses method of content delivery in a content-delivery network comprising at least a tier of edge servers (*first tier*). Ex-1003, ¶¶215-219.

b. 1[a]

Swildens discloses or suggests this limitation.

Swildens discloses that "[a]n edge server receives a user's request for content" (Swildens, [0046], [0068]), among a group of edge servers (*first tier*), (*see, e.g.*, *id.*, [0009] ("FIG. 2 illustrates…a user accessing popular content ***from a group of edge servers***"), [0044]-[0045], Fig. 2); Ex-1003, ¶222.

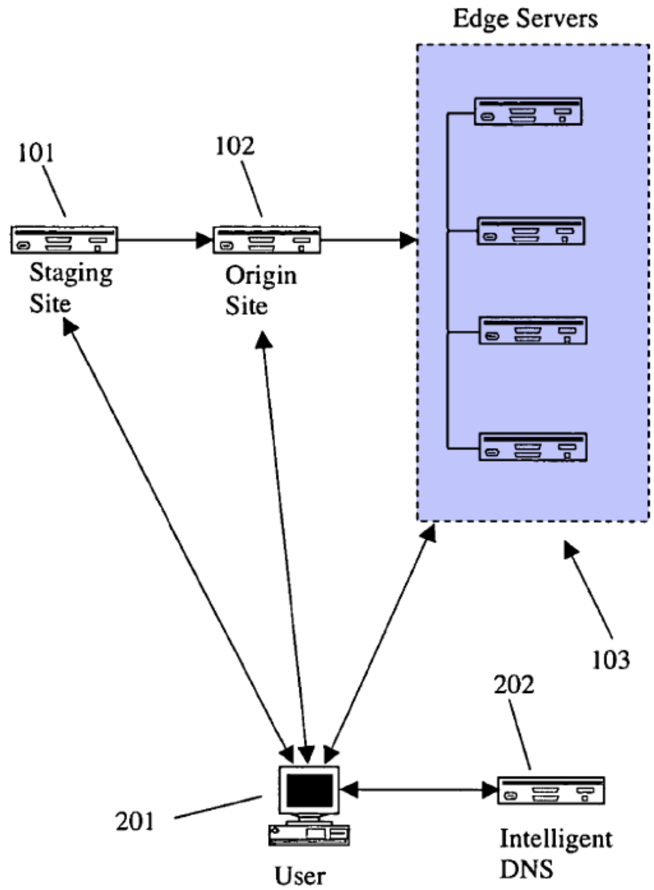


Fig. 2

Swilden, Fig. 2

Swildens also discloses that "a customer…select[s] content files that are to be transferred to a group of edge servers," explaining that the customer uploads the content files to the network through a staging server, and one or more origin servers "store all of a customer's available content files." Swildens, [0042]; *id.*, [0005] (disclosing access to "***an information provider's collection*** of multimedia files"), [0009] (defining a "customer" as "[t]he provider of a subset of ***an entire content***

collection"). A POSITA would have understood that the content file(s) requested comes from a collection (*library*) of content uploaded by a customer for distribution over the network. Ex-1003, ¶223.

Accordingly, Swildens discloses an edge server (*first server*) among a group of edge servers (*first tier*), obtaining a request from a client for content (*resource*) available as part of a content provider's collection (*library*). Ex-1003, ¶¶220-223.

c. 1[b]

Swildens, alone or combined with Seed, teaches or suggests this limitation.

Swildens discloses that "[a]n edge server maintains a dynamic number of popular files in its memory for the customer." Swildens, [0043], [0055]. Upon receiving a request for a content file, "[i]f the edge server has the requested file, it then forwards or streams the file to the user 201 and records a hit for the file. If the edge server does not have the file, it records the miss in its running count and redirects the user 201 to the origin site 102." *Id.*, [0068]; *see also id.*, [0046]. Swildens also discloses that "[t]he edge server checks the frequency of requests over a certain amount of time…to see if any new files have gained enough popularity to replace another file in the local storage." *Id.*, [0068]. In other words, Swildens discloses that if the resource is not available at the first edge server, the edge server records a miss and then determines whether the missing file is popular by evaluating the frequency of requests for the missing file compared to other files. Ex-1003, ¶¶224-226.

If this limitation requires determining whether the resource is popular directly in response to determining the resource is missing from the first server—and not just at any subsequent time—Swildens would have been understood as disclosing this as well. Ex-1003, ¶227. Although Swildens checks the popularity of a missing file "over a certain amount of time (configurable by an administrator)," a POSITA would have understood that this refers to the window of time the edge server uses to determine popularity (e.g., number of requests/minute), and not an interval at which the server determines popularity (e.g., evaluate popularity every 5 minutes). Ex-1003, ¶¶227-229; *compare, e.g.*, Swildens, [0068] (after a miss, evaluating "the frequency of requests over a certain amount of time"), *with id.*, [0126] (describing another embodiment that evaluates popularity "at some specified time interval" using a centralized "auto migration server"). In other words, a POSITA would have understood that Swildens discloses or at least suggests determining the popularity of a missing file in response to determining the file is missing. Ex-1003, ¶230.

But even if Swildens cannot be read this way, configuring Swildens to do so would have been obvious in view of Seed. Seed teaches that "[i]f the selected edge server does not have the requested object, a check is initiated (at 315) for the edge server to determine whether the requested object is popular." Seed, [0026], Fig. 3(a); Ex-1003, ¶231.

A POSITA would have been motivated to configure Swildens' edge servers to determine the popularity of a missing file directly in response to determining the file is missing to ensure edge servers update their caches immediately after determining popular files are missing. Ex-1003, ¶¶232-233. This would both provide better quality of service to users requesting files that rapidly become popular, and reduce the number of redirects to other servers. *Id*.

A POSITA would have had a reasonable expectation of success in making this combination because any modification would simply require reconfiguring what triggers the edge server to make a popularity determination—finding a file is missing consistent with Seed compared to, e.g., some other trigger or interval. Ex-1003, ¶234. Indeed, this modification would have merely involved computer code that a POSITA would be readily capable of implementing. *Id*.

d. 1[c]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed teaches or suggests this limitation.

Swildens discloses that "[w]hen a file has been requested from an edge server a sufficient number of times to become more popular than the lowest popular stored file, then the file is obtained from the origin site 102" (*if the resource is determined* *to be popular, then the first server obtaining the resource*). Swildens, [0055], [0068]; Ex-1003, ¶237.

Swildens discloses that, in response to a first request for a popular but not-yet-obtained file, the client should be redirected to a server that has the file (*id.*, [0068]-[0069]), suggesting the file is obtained by the edge server later. Ex-1003, ¶238. However, a POSITA would have understood that clients routinely request delivery of the same object multiple times (e.g., by revisiting the same website or requesting the same content again). Ex-1003, ¶239. Thus, for any subsequent client-requests for the file after the file has been obtained, the edge server would serve the object to the client rather than redirect to another server (*the first server serving the resource to the client*). Ex-1003, ¶239; *Smith*, 276 F.3d at 1311.

If, however, the claim requires the edge server serve the object to the client after only a first request, a POSITA would have been motivated to configure Swildens' edge servers to fetch popular content rather than redirect clients to the popular content in view of Seed. Ex-1003, ¶240. Seed teaches that conventional CDNs routinely have edge servers configured to "fetch objects" from other servers "when a requested object is not available on a handling edge server" to serve the object from the edge server that received the request. Seed, [0013]; Ex-1003, ¶241. Although Seed prefers redirecting a first request to a different server like Swildens

(Seed, [0013]-[0014]), Seed nonetheless describes its network as being configured to fetch objects, (*see, e.g.*, Seed, [0046], [0059]; Ex-1003, ¶241).

Moreover, there were only three typical ways for a server to ensure that requests for an object missing from its cache are fulfilled: (1) fetch-and-server; (2) redirect the client; or (3) forward the request, each of which would have been obvious to try. §VI.A.2.d (Ground 1, 1[c]); Ex-1003, ¶¶242-243; Ex-1017, 2:38-42; Ex-1018, [0040]; *Perfect Web Techs.*, 587 F.3d at 1331. Alternatively, replacing Swildens' redirecting-and-fetching with fetching-and-serving would have merely amounted to a simple substitution of one known element (redirecting-and-fetching) for another (fetching-and-serving), or replacing one known technique (redirecting-and-fetching) with another suitable option (fetching-and-serving) to achieve a predictable result or solve a known problem in the art (serving requested but missing files). Ex-1003, ¶243; *Intel*, 61 F.4th at 1380; *Snap*, 2025 WL 2793538, *3-4.

A POSITA would have also been specifically motivated to replace Swildens' redirect-and-fetch process with a fetch-and-serve process because fetching provides different benefits. Ex-1003, ¶244; *Gurley*, 27 F.3d at 553; *General Electric*, 983 F.3d at 1352. For instance, fetching would reduce CDN-processing and transmission costs by avoiding having to transfer the same object twice from another server—once to the client, and again to the edge server replicating the object later. Ex-1003, ¶244; Swildens, [0043].

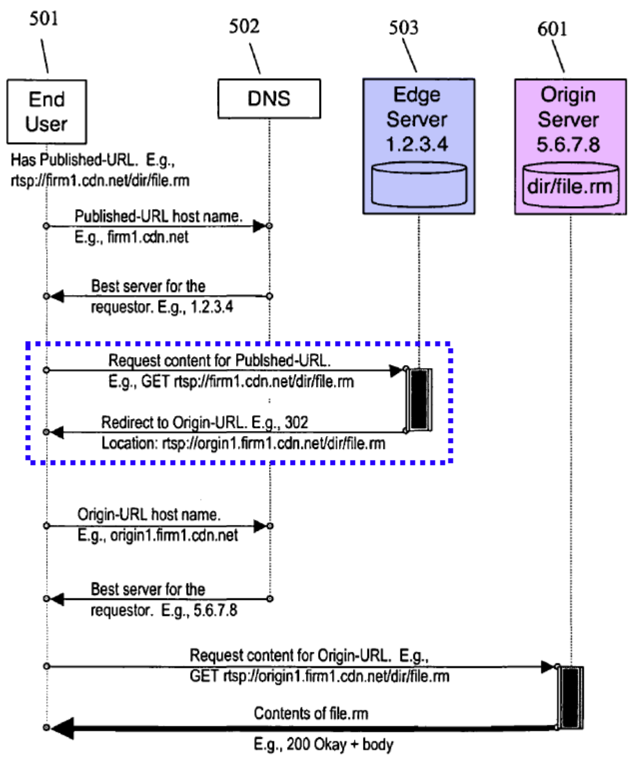
A POSITA would have also had a reasonable expectation of success in making this modification because both fetching and redirecting were well-known, and swapping one for another merely involves changing which network system obtains the requested object; rather than have an edge server tell the client where to look, the edge server would fetch the object itself from that location (which it is already programmed to do, only ***after*** redirecting first) and then provide it to the client. Ex-1003, ¶245; Swildens, [0055], [0068]. The minimal changes to computer code needed to make this modification would have been well within the skills of a POSITA to implement. Ex-1003, ¶245.

Accordingly, Swildens alone or combined with Seed teaches or suggests that if the file (*resource*) is popular, the edge server (*first server*) obtains the file and serves it to the client either in response to a subsequent request or as part of a fetch-and-serve process. Ex-1003, ¶¶235-245.

e. 1[d][i]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed teaches or suggests this limitation.

Swildens discloses that an edge server "maintains a dynamic number of popular files in its memory." Swildens, [0043]. Swildens discloses that when the edge server does not have the requested file—because the file either is not popular, or the edge server has not yet determined that the file is popular—the server "redirects the user 201 to the origin site 102" among the origin group of servers. Swildens, [0068], Fig. 6; §VI.B.1.a (1[pre]); Ex-1003, ¶¶246-247.



Delivery of unpopular content
Fig. 6

Swildens, Fig. 6

Accordingly, Swildens discloses if the resource is determined not to be popular, directing the client to an intermediate-origin server (*second server*) in an intermediate origin group of servers (*second tier*) distinct from the edge-level servers (*first tier*). Ex-1003, ¶248.

Although Swildens alone meets this limitation, Swildens combined with Seed also meets this limitation by mapping a different set of servers (e.g., "parent" servers) to the second tier of servers. Ex-1003, ¶¶249-251. Specifically, a POSITA would have been motivated to incorporate an additional layer of servers—and specifically "parent" servers—between Swildens' edge and intermediate-origin servers when designing a network based on tradeoffs in cost and performance, or replace Swildens' intermediate-origin tier with a parent tier entirely. *Id*. Swildens, for instance, recognizes that deciding how many servers to include in a network and how they are arranged relative to one another is a design choice based on cost and performance factors. Swildens, [0034], [0044]-[0045], [0077]-[0078], [0095]-[0101]; Ex-1003, ¶249.

Although Swildens only mentions two types of servers in its network—edge and origin—those in the art routinely incorporated a third "parent" type between edge and origin servers, often strategically located in geographic regions near the edges to minimize load on the origin server(s), minimize latency, and maximize quality of server. Ex-1003, ¶250; Ex-1019, 1:12-64, 4:63-5:21, Fig. 3. Seed, for instance, expressly teaches another CDN that performed nearly identical functions as Swildens—storing popular content on edge servers, and unpopular content on intermediate servers (*compare, e.g.*, Swildens, [0055], [0068], *with* Seed, [0016], [0025]-[0028]; Ex-1003, ¶250), but stores the unpopular content on an intermediate tier of "parent" servers in addition to "origin" servers like Swildens. *Compare, e.g.*, Swildens, [0055], [0068], *with* Seed, [0016], [0025]-[0028]; Ex-1003, ¶250.

A POSITA would have been motivated to either implement an additional intermediate tier of “parent” servers between Swildens’ origin and edge servers, or replace Swildens’ edge servers entirely, and distribute distinct portions of unpopular content across the group of parent servers, consistent with Seed. Ex-1003, ¶¶249-251. Adding an additional tier of parent servers would improve CDN performance by providing caches of unpopular content closer to the edge, and allowing requests to be distributed across more servers. Ex-1003, ¶250. And replacing Swildens’ intermediate-origin tier with an intermediate-parent tier would be less costly to maintain, while providing near-identical operations that Swildens already discloses, meaning it is merely a simple substitution of one known element (intermediate-origin servers) with another (intermediate-parent servers) to obtain a predictable result (storing and serving not popular content). Ex-1003, ¶251.

A POSITA would have had a reasonable expectation of success in implementing an additional intermediate tier of “parent” servers consistent with Seed not only because implementing parent servers in a network was well-known, but also because Swildens already envisions adjusting the number and arrangement of servers based on design considerations, and Seed provided an express framework to structure these servers to accomplish the same goals as Swildens of storing popular content close to the edge. Ex-1003, ¶252.

Upon making this combination, Swildens combined with Seed teaches that if the resource is determined not to be popular, directing the client to an intermediate-parent server (*second server*) in an intermediate parent group of servers (*second tier*) distinct from the edge-level servers (*first tier*). Ex-1003, ¶253.

f. 1[d][ii]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed teaches or suggests this limitation.

Swildens discloses that although its intermediate origin "servers together store the complete copies of all the content for a particular customer" (Swildens, [0050]), "[a]n individual origin server does not have to be able to hold all of the content" (*id.*, [0060]). Instead, Swildens discloses using a "hash function"—an algorithm to "map[] the large number of content identifiers across the much smaller number of storage servers" (*id.*, [0061])—to distribute distinct portions of a content provider's library across a group of origin servers and identify on which server any given content resides (*id.*, [0060]; Ex-1003, ¶¶254-255). Unless configured specifically to allow overlap for redundancy (which Swildens recommends for improving performance, but does not require, Swildens, [0058]), a POSITA would have understood that a hash function would typically result in each server having distinct, non-overlapping portions of the library, because each content identifier would be assigned to one server based on its hash value. §V (Claim Construction); Ex-1003, ¶255.

Accordingly, Swildens discloses an intermediate origin server (*second server*) stores one portion of the content provider's library that would include the requested file (*resource*), but other intermediate origin servers (*other servers*) in the group of origin servers (*second tier*) would store other, distinct portions of the content provider's library. Ex-1003, ¶256.

Upon implementing an additional or alternative intermediate tier of parent servers consistent with Seed (§VI.B.1.e (1[d][i])), Swildens and Seed together teach an intermediate parent server (*second server*) stores one portion of the content provider's library that would include the requested file (*resource*), but other intermediate parent servers (*other servers*) in the group of parent servers (*second tier*) would store other, distinct portion of the content provider's library. Ex-1003, ¶257.

g. 1[d][iii]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed teaches or suggests this limitation.

Swildens discloses that the group of origin servers (*second tier*) are intermediate to the group of edge servers (*first tier*) and the customer/staging server that stores files associated with the content provider's library. §§VI.B.1.a-VI.B.1.b

(1[pre]-1[a]); Ex-1003, ¶¶258-259. A POSITA would have understood that Swildens' customer/staging server is another type of "origin server" (e.g., the content provider's) consistent with the '692 patent. Ex-1003, ¶259.

Specifically, the '692 patent recognizes—consistent with convention—that a network involving a CDN may have two different tiers of origin servers: (1) a customer-provided origin server, and (2) a CDN-provided origin server intermediate to the customer-provided origin server and edge servers. Ex-1001, 3:65-4:17, Fig. 3; *see also* Ex-1003, ¶¶128, 259; Ex-1019, 1:19-46.

A POSITA would have therefore understood that Swildens' customer/staging server is consistent with the '692 patent's customer-provided "origin server," and Swildens' "origin server" is consistent with the '692 patent's intermediate, CDN-provided "origin server." Ex-1003, ¶¶126-130, 138, 259; *compare* Swildens, [0054] ("A customer selects multimedia files that are to be transferred to the edge servers," which are provided "from the customer site 100"), Fig. 1, *with* Ex-1001, 3:65-4:1 ("A content provider's/customer's server (or servers) are also referred to as origin servers," which "may be owned and/or operated by that content provider"), Figs. 3, 4; *see also* Swildens, [0056] ("The customer is charged for the amount of storage required on the origin site 102 to store the customer's files…. This allows the customer to adjust the amount of storage anticipated for file demand," meaning the customer does not own Swildens' "origin" servers).

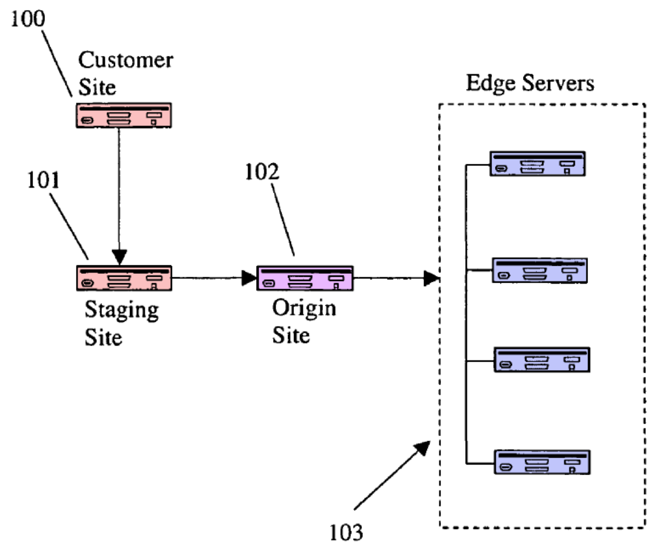
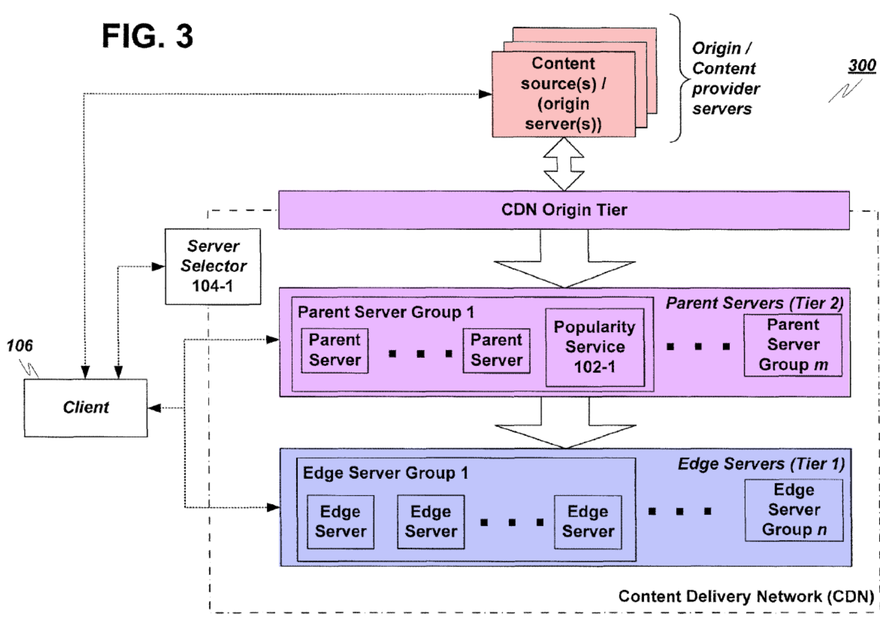


Fig. 1

Swildens, Fig. 1

FIG. 3



Ex-1001, Fig. 3

But even if Swildens' customer/staging server cannot be considered an "origin" server, upon implementing an additional or alternative intermediate tier of parent servers consistent with Seed (§VI.B.1.e (1[d][i])), Swildens and Seed together teach a parent group of servers (*second tier*) intermediate to the edge group of servers (*first tier*) and Swildens' origin servers that store files associated with the content provider's library. Ex-1003, ¶260.

h. 1[d][iv]

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed discloses or suggests this limitation.

Swildens discloses the origin server (*second server*) serves unpopular or yet-to-be-replicated files to the client. Swildens, [0069] ("If the user is redirected to the origin site 102, the origin site 102 checks its local storage for the requested file. If the origin site 102 has the file, it sends or streams the file to the user 201."), [0047] (same), [0102] ("The origin server can have the performance characteristics to deliver relatively unpopular content or content that has just become popular, but not yet replicated by the invention."); Ex-1003, ¶¶261-262.

Alternatively, upon implementing an additional or alternative intermediate tier of parent servers consistent with Seed (§VI.B.1.e (1[d][i])), a POSITA would have been motivated and had a reasonable expectation of success in configuring the parent server (*second server*) to serve the unpopular or yet-to-be-replicated files to the client, consistent with Seed. Seed, [0028]-[0029], Fig. 3(a); Ex-1003, ¶263.

2. Claim 2

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed discloses or suggests this additional limitation.

Swildens discloses partitioning unpopular content across an intermediate tier of servers using a hash function, and specifically "comput[ing] a hash…from a content file identifier" and storing the file at an intermediate tier based on the hash. Swildens, [0051]-[0052]. Swildens also teaches that "[a] good hash function is one that spreads the content storage requirements of the content identifiers across the actual storage capacities of each of [intermediate] server group." *Id.*, [0061].

Accordingly, Swildens discloses partitioning a content provider's library across intermediate-origin servers (*second tier*) using content-file identifiers and storage capacity of each intermediate server to partition (*logic*). Ex-1003, ¶¶264-266.

Additionally, Swildens combined with Seed teaches partitioning a content provider's library across intermediate-parent servers (*second tier*) using content-file identifiers and storage capacity of each intermediate server to partition (*logic*). §VI.B.1.e (1[d][i]); Ex-1003, ¶267.

3. Claim 4

Swildens discloses or suggests this additional limitation.

Swildens discloses the "multimedia files" distributed over the network include (among other things) "music and movies." Swildens, [0003]; *see also id.*, [0129] ("popular content can vary from one edge location to another. For example, the top ten songs in the Atlanta area will be different from the top ten songs in London."); Ex-1003, ¶¶268-269. A POSITA would have understood "music and movies" are "content types" as claimed, and that each file would be associated with one of the content types. Ex-1003, ¶269.

4. Claim 5

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed discloses or suggests this additional limitation.

Swildens discloses storing distinct portions of a content provider's library across multiple different intermediate-origin servers (*second tier*). §§VI.B.1.e-VI.B.1.g (1[d][i]-[iii]); Ex-1003, ¶¶270-271.

Additionally, Swildens combined with Seed teaches storing distinct portions of a content provider's library on multiple different intermediate-parent servers (*second tier*). §§VI.B.1.e-VI.B.1.g (1[d][i]-[iii]); Ex-1003, ¶272.

Swildens also discloses the library may include multiple content types, including music (audio) and movies (video) content. §VI.B.3 (cl. 4); Ex-1003, ¶273.

A POSITA would have appreciated that the file content types would be distributed across the intermediate servers—whether origin or parent—by some proportion (e.g., some intermediate servers would have some proportion of objects of a certain content type). Ex-1003, ¶274. Indeed, distributing content across servers by type was known as a means to balance load. *Id.* (citing Ex-1020, [0038] (spreading different video files over different servers); Ex-1021, 8:25-56).

5. Claim 6

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed teaches or suggests this additional limitation.

Swildens explains that "[a] good hash function is one that spreads the content storage requirements of the content identifiers across the actual storage capacities of each [intermediate] server group." Swildens, [0061]. As such, a POSITA would have appreciated that when Swildens is implemented in a network with intermediate (origin or parent, §VI.B.1.e (1[d][i])) servers that have the same amount of storage, Swildens' hash function would result in a substantially equal distribution of content across the parent-level servers (*servers in the second tier*). Ex-1003, ¶¶275-277.

Moreover, Swildens further explains that hash values are determined based on "content identifiers" or "published URLs" associated with the content (Swildens, [0061]-[0062], [0065]-[0066]), and "[r]edirected requests to [intermediate] servers have a hash included in the URL" that is used to locate which intermediate server is storing the requested object, (Swildens, [0060]). As such, Swildens discloses distributing content across the intermediate-level servers (*servers in the second tier*) based on hash values either included in URLs, or of URLS themselves (*based on hash values of uniform resource locators (URLs)*) associated with content via content identifiers or their published URLs (*associated with two or more respective resources in the content provider's library*). Ex-1003, ¶278.

6. Claim 7

If both 1[c] and 1[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed discloses or suggests this additional limitation.

Swildens discloses directing the client to an intermediate-origin server (*second server*) in an intermediate-origin tier (*second tier*) using a hash to determine which intermediate-origin server (*second*) in the tier (*second tier*) stores the portion of the library with the requested file (*resource*). §§VI.B.1.e-VI.B.1.g (1[d][i]-[iii]); Ex-1003, ¶¶279-280.

Alternatively, Swildens combined with Seed teaches directing the client to an intermediate-parent server (*second server*) in an intermediate-parent tier (*second tier*) using a hash to determine which intermediate-parent server (*second*) in the tier (*second tier*) stores the portion of the library with the requested file (*resource*). §§VI.B.1.e-VI.B.1.g (1[d][i]-[iii]); Ex-1003, ¶281.

7. Claim 8

Swildens discloses or suggests this additional limitation.

Swildens discloses that "when a file has been requested from an edge server a sufficient number of times to become more popular than the lowest popular stored file, the file is obtained from an origin site." Swildens, [0043], [0055], [0068]. A POSITA would have understood that the request count or frequency of the "lowest popular stored file" serves as a "popularity threshold" as claimed for each edge server. Ex-1003, ¶284.

However, Swildens also discloses another embodiment of the network that uses a "miss count threshold value (e.g., ten misses in one hour)" to determine whether a requested (but missing) file is popular. Swildens, [0123]-[0133]. Swildens explains that using a "miss count threshold value" can reduce the number of files stored on edge servers by setting the threshold to an appropriate value. *Id.*, [0131]-[0132]; Ex-1003, ¶285.

A POSITA would have been motivated to apply a similar "miss count threshold value" to popularity determinations made at edge servers instead of Swildens' rank-order threshold to both reduce transmission costs to update the edge server (by limiting how many miss counts are required to be deemed popular), and provide users the ability to better control how many popular files would be stored at the edge. Ex-1003, ¶¶286-289; Swildens, [0043] ("The customer is charged for the bandwidth used to transfer files from the origin servers to the edge servers.").

Accordingly, Swildens discloses determining whether the resource is popular includes determining whether a current rank (*popularity value*) for a file (*resource*) exceeds either the lowest rank (*popularity threshold*) of the files stored at the edge. Ex-1003, ¶¶282-289. Alternatively, Swildens renders obvious determining whether the resource is popular includes determining whether a "miss count" (*popularity value*) for a file (*resource*) exceeds a miss-count threshold value (*popularity threshold*). *Id.*

8. Claim 10

a. 10[pre]

Swildens discloses or suggests this limitation.

Swildens discloses or suggests a content-delivery framework, including a CDN and method for using one. *See* §VI.B.1.a (1[pre]); Ex-1003, ¶¶290-291.

a. 10[a]

Swildens discloses or suggests this limitation.

Swildens discloses or suggests a plurality of edge servers (*first tier*). *See* §VI.B.1.a (1[pre]); Ex-1003, ¶¶292-293.

b. 10[b]

Swildens combined with Seed discloses or suggests this limitation.

Swildens combined with Seed discloses or suggests a plurality of parent servers (*second tier*). *See* §VI.B.1.e (1[d][i]); Ex-1003, ¶¶294-295.

c. 10[c]

Swildens discloses or suggests this limitation.

Swildens discloses that rather than solely having each edge server make its own popularity determination, in an alternative embodiment, "an auto migration server [(*popularity service*)] collects all of the log files…from the edge servers [(*constructed and adapted to obtain a first set of information from…the edge servers*)]…and tabulates the miss count…for every piece of content" (*the popularity service making its determinations based…on a measure of popularity of the requested content*) to determine whether any content is sufficiently popular based on a threshold such that it should be auto-migrated to the edge (*provide the edge servers with a second set of information about which tier of servers should handle those requests for content*). Swildens, [0126], [0128], [0130]; Ex-1003, ¶¶296-297.

d. 10[d]

Swildens combined with Seed discloses or suggests this limitation.

Swildens discloses or suggests that the requested content is available as part of a content provider's library, (§VI.B.1.b (1[a])), and Swildens combined with Seed teaches or suggests storing distinct portions of the content provider's library that are partitioned using a hash function (*logically*) across parent-level servers (*second tier*). §§VI.B.1.e-VI.B.1.f (1[d][i]-[ii]), VI.B.2 (cl. 2); Ex-1003, ¶¶298-299.

9. Claim 11

Swildens combined with Seed teaches or suggests this limitation.

Swildens combined with Seed teaches or suggests storing distinct portions of the content provider's library that are partitioned using a hash function (*logically*), including a first portion with the requested content, on one parent server in the parent-level servers (*second tier*). §§VI.B.1.e-VI.B.1.f (1[d][i]-[ii]), VI.B.2 (cl. 2); Ex-1003, ¶¶300-301.

10. Claim 12

Swildens combined with Seed teaches or suggests this limitation.

Swildens combined with Seed teaches or suggests storing distinct portions of the content provider's library that are partitioned using a hash function (*logically*), including a second portion on another parent server among the parent-level servers (*second tier*) distinct from the first portion. §§VI.B.1.e-VI.B.1.f (1[d][i]-[ii]), VI.B.2 (cl. 2); Ex-1003, ¶¶302-303.

11. Claim 13

Swildens combined with Seed teaches or suggests this limitation.

Swildens combined with Seed teaches or suggests incorporating a tier of parent servers into Swildens' CDN, and using a hash function (*logic*) to partition and then identify which of a plurality of parent servers (*second tier*) is storing a portion of a content provider's collection (*library*). §§VI.B.1.e-VI.B.1.f (1[d][i]-[ii]), VI.B.2 (cl. 2), VI.B.8.d (10[d]); Ex-1003, ¶¶304-305. Swildens also discloses using a central server to make collective popularity determinations for a plurality of edge servers. §VI.B.8.c (10[c]); Ex-1003, ¶305.

Upon incorporating a parent tier of servers into Swildens' system, a POSITA would have appreciated that the central server would send to edge servers information that maps the content identifiers to the parent server in a parent level of servers that would store the portion of the content provider's collection (library) that stores the requested content. *See* Swildens, [0065]; Ex-1003, ¶306.

12. Claim 14

Swildens combined with Seed discloses or suggests this limitation.

Swildens combined with Seed teaches or suggests the parent-level servers are intermediate between the edge servers (*first tier*) and an origin server that stores the content provider's library. §§VI.B.1.e (1[d][i]), VI.B.1.g (1[d][iii]); Ex-1003, ¶¶307-308.

13. Claim 15

a. 15[pre]

If limiting, Swildens discloses or suggests this limitation.

Swildens discloses a method of content delivery in a CDN comprising edge and origin levels (*a plurality of tiers*), each with a plurality of servers, including edge servers (*first tier*) and origin servers (*second tier*). §VI.B.1.a (1[pre]); Ex-1003, ¶¶309-310.

b. 15[a]

If limiting, Swildens discloses or suggests this limitation, which may be mapped in at least two ways based on dependent claims 17 and 18. Specifically, dependent claims 17 and 18 both specify the first-tier servers are edge servers and the second-tier servers are parent servers. But dependent claim 17 specifies the first server is an ***edge*** server. and dependent claim 18 specifies the first server is a ***parent*** server. Ex-1003, ¶¶311-312.

For dependent claim 17, Swildens discloses an edge server (*first server*) among a group of edge servers (*first tier*) obtaining a request from a client for a file (*resource*) available as part of a content provider's library. §VI.B.1.b (1[a]); Ex-1003, ¶313.

For dependent claim 18, Swildens combined with Seed teaches a parent server (*first server*) among a group of parent servers (*second tier*) obtaining a request from a client for an file (*resource*) available as part of a content provider's library. Swildens, [0068] (redirecting request to intermediate-origin servers), §VI.B.1.e (1[d][i]) (implementing intermediate-parent servers between edge and origin servers configured like Swildens' origin servers); Ex-1003, ¶314.

c. 15[b]

If 15[b] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed discloses or suggests this limitation, which may mapped in several ways based on dependent claim 15 alone, or in view of claims 17 and 18. Ex-1003, ¶¶315-316.

Swildens discloses that if the file (*resource*) is not available at the edge server (*first server*), the request from the client is redirected to an intermediate-origin server (*second-tier server*) distinct from the group of edge servers (*first tier*) of the initial edge server (*first server*). §§VI.B.1.a (1[pre]), VI.B.1.e (1[d][i]); Ex-1003, ¶317.

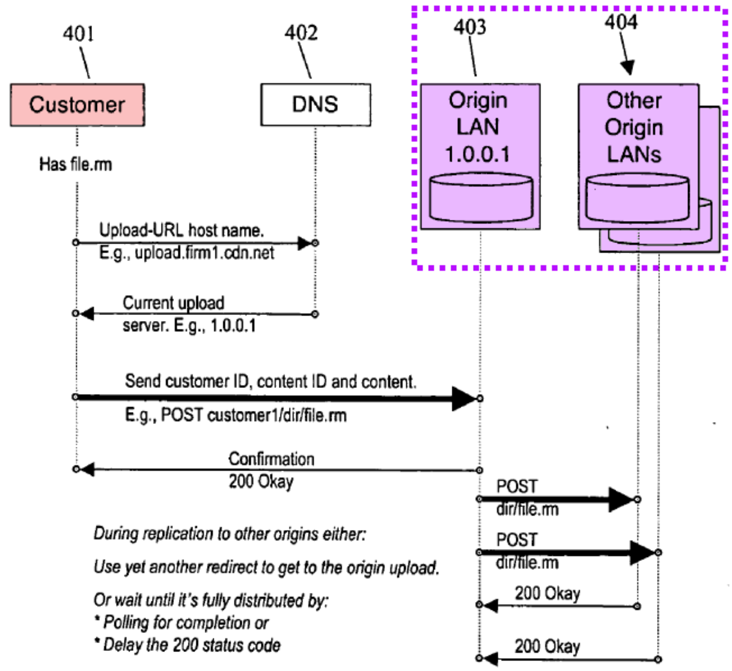
For dependent claim 17, however, Swildens combined with Seed also teaches that if the file (*resource*) is not available at the edge server (*first server*), the request from the client is redirected to an intermediate-parent server (*second-tier server*) distinct from the group of edge servers (*first tier*) of the initial edge server (*first server*). §VI.B.1.e (1[d][i]); Ex-1003, ¶318.

For dependent claim 18, this limitation is contradictory for the reasons explained in Ground 1, and at least three interpretations may apply. §VI.A.15.c (Ground 1, 15[b]); Ex-1003, ¶¶319-320. But regardless of which interpretation applies, Swilden combined with Seed teaches each. *Intel*, 21 F.4th at 813.

(1) Swildens discloses that "[a]n individual origin server does not have to be able hold all of the content on its local storage device" and "is able

to use to determine which other of the LAN-connected origin servers in its group should hold the content on its local storage device." Swildens, [0060]. Moreover, the origin server that a hash function points to may not necessarily have the requested content (e.g., if an intermediate server changes its cache for some reason), meaning a further redirect would be required. Swildens, [0046] (noting redirection to an intermediate "server that ***should*** have the content"); Ex-1003, ¶¶321-322. When incorporating an additional or alternative tier of parent servers to perform the functions Swilden assigns to the origin servers, however (§VI.B.1.e (1[d][i])), Swildens combined with Seed suggests redirecting a request from a parent server (*first server*) among parent servers (*second tier*) to at least another parent server also among the parent servers (*second tier*). Ex-1003, ¶323; *see also* Seed, [0029] (redirecting between parents).

(2) Swildens also envisions multiple layers of origin servers, particularly when using an "origin" server to serve the role of a "staging" server. *See, e.g.*, Swildens, Fig. 4, [0121].



Content upload and distribution to origins

Fig. 4

Seed likewise envisions providing multiple layers of parent servers. Seed, [0022]. When incorporating an additional or alternative tier of parent servers to perform the functions Swildens assigns to the origin servers (§VI.B.1.e (1[d][i])), it would have been obvious to redirect a request from one level of parent servers (*second tier*) to a distinct level of parent servers (a distinct *second tier*), consistent with Seed. Ex-1003, ¶¶324-325. For instance, a POSITA would have recognized that one layer of parent servers closer to the edge may be used to store objects that are less popular than those at the edge, but more popular than those stored by other lower-tier parent servers further from the edge. Ex-1003, ¶325.

(3) Finally, Swildens discloses that "[i]f the origin site 102 does not have the file, then it redirects the user to the staging site 101," which is a distinct staging-level (*tier*) from the origin servers. Swildens, [0069], §VI.B.1.a (1[pre]); Ex-1003, ¶326. When incorporating an additional or alternative tier of parent servers to perform the functions Swildens assigns to the origin servers, however (§VI.B.1.e (1[d][i])), a POSITA would have appreciated that "where the parent servers [(*first server in the second tier*)] collectively are not populated with all of the objects and the network has access to the origin server of a requested object, the client may be redirected (at 225, 320) to the origin server (*second server*) if the requested object is not available on the parent servers," consistent with Seed. Seed, [0031]; Ex-1003, ¶326.

Under any possible interpretation, Swildens alone (with its staging, origin, and edge tiers), or Swildens combined with Seed (with a combined staging, origin, parent, and edge tiers), discloses or suggests this limitation. Ex-1003, ¶327. Moreover, under any interpretation, the redirecting would be "based, at least in part, on a current popularity value for the resource" because Swildens discloses that the origin servers (or parent, in view of Seed) redirect requests only *after* the edge server has determined an object is not popular. Swildens, [0046]; Ex-1003, ¶327.

d. 15[c]

If 15[c] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed teaches or suggests this limitation.

For both dependent claims 17 and 18, Swildens alone or combined with Seed discloses partitioning the content provider's library across an intermediate group of servers (origin or parent) (*second tier servers*), such that each intermediate server (origin or parent) stores distinct portions of the content provider's library, including one server with one portion that would have the requested object (*resource*), and a separate server with a distinct portion, with the servers (*second tier)* being intermediate to the edge-level servers (*first tier*) and a staging/origin server that stores the content provider's library. §§VI.B.1.f-VI.B.1.g (1[d][ii]-[iii]); Ex-1003, ¶¶328-330.

e. 15[d]

If 15[d] must be performed (§V (Claim Construction)), Swildens alone or combined with Seed teaches or suggests this limitation.

For both dependent claims 17 and 18, Swildens alone or combined with Seed discloses redirecting a request from the client to an intermediate server (origin or parent) (*second-tier server*) by determining to which server (origin or parent) the request should be redirected using a hash function to identify the server that stores the portion of the content provider's library with the requested file. §§VI.B.1.e-VI.B.1.f (1[d][i]-[ii], redirecting), VI.B.6 (cl. 7, hash function); Ex-1003, ¶¶331-333.

14. Claim 17

Swildens combined with Seed teaches or suggests the method of claim 15, in which the first-tier servers are edge servers, the second-tier servers are parent servers, the first server is an edge server, and the second server is a parent server. §§VI.B.13.a-VI.B.13.e (15[pre]-[d]); Ex-1003, ¶¶334-335.

15. Claim 18

Swildens combined with Seed teaches or suggests the method of claim 15, in which the first-tier servers are edge servers, the second-tier servers are parent servers, the first server is a parent server, and the second server is either another parent server (in the same or a distinct level of parent servers) or an origin server. §§VI.B.13.a-VI.B.13.e (15[pre]-[d]); Ex-1003, ¶¶336-337.

VII. MANDATORY NOTICES

A. Real Party-in-Interest

The Petitioner and real party-in-interest is Microsoft Corporation.

B. Related Matters

Sandpiper asserted the '692 patent in the following litigations:

- *Sandpiper CDN, LLC v. Microsoft Corp.*, No. 2-25-cv-00664 (E.D. Tex., June 26, 2025);

- *Sandpiper CDN, LLC v. Google LLC*, No. 2:24-cv-03951 (N.D. Cal., May 10, 2024); and
- *Sandpiper CDN, LLC v. Comcast Cable Commc'ns Mgmt. LLC*, No. 2:24-cv-00886 (E.D. Tex., Nov. 1, 2024).

C. Lead and Back-Up Counsel, and Service Information

I. LEAD COUNSEL
Jessica Kaiser (Reg. No. 58,937) kaiser-ptab@perkinscoie.com PERKINS COIE LLP 1900 Sixteenth Street, Suite 1400 Denver, CO 80202-5255 Telephone: (303) 291-2300
BACK-UP COUNSEL
Christopher Marando (Reg. No. 67,898) Marando-ptab@perkinscoie.com PERKINS COIE LLP 700 Thirteenth Street N.W., Suite 800 Washington, DC 20005-3960 Phone: (202) 654-6200 Matthew A. Lembo (Reg. No. 75,633) Lembo-ptab@perkinscoie.com PERKINS COIE LLP 1155 Avenue of the Americas, 22nd Floor New York, NY 10036-2711 Telephone: (212) 262-6900

Steven Beigelmacher (Reg. No. 75,050)
beigelmacher-ptab@perkinscoie.com
PERKINS COIE LLP
1301 Second Avenue, Suite 4200
Seattle, WA 98101-3804
Telephone: (206) 359-8000

Sarah Fowler (Reg. No. 54,854)
stahnke-ptab@perkinscoie.com
PERKINS COIE LLP
3150 Porter Drive
Palo Alto, CA 94304
Telephone: (650) 838-4300

Petitioner consents to electronic service at the following email address: Microsoft-Sandpiper-IPR@perkinscoie.com.

VIII. CONCLUSION

Petitioner requests institution of *inter partes* review and cancellation of the Challenged Claims.

Respectfully submitted,

/ *Jessica Kaiser* /

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

Dated: December 17, 2025

CERTIFICATE OF COMPLIANCE

The undersigned hereby certifies that the foregoing **Petition for *Inter Partes* Review** contains 13,987 words, excluding those portions identified in 37 C.F.R. §42.24(a), as measured by the word-processing system used to prepare this paper.

Respectfully submitted,

/ *Jessica Kaiser* /

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

Dated: December 17, 2025

CERTIFICATE OF SERVICE

The undersigned certifies that the foregoing Petition for *Inter Partes* Review of U.S. Patent No. 9,762,692, the associated Power of Attorney, and Exhibits Ex-1001 through Ex-1022 were served on December 17, 2025 by FedEx Priority Overnight® on the correspondence address of record indicated in the Patent Office's Patent Center system for U.S. Patent No. 9,762,692.

STERNE, KESSLER, GOLDSTEIN & FOX P.L.L.C.
1101 K Street, NW, 10th Floor
WASHINGTON, DC 20005

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

Andrea Leigh Fair - andrea@millerfairhenry.com
Carolyn E Drell - cdrell@bsfllp.com
Connor Coupe - ccoupe@bsfllp.com
Garrett C. Parish - garrett@millerfairhenry.com
Mark D. Schafer - mschafer@bsfllp.com
Philip Eckert - peckert@bsfllp.com
Ryan D. Dykal - rdykal@bsfllp.com

Respectfully submitted,

/ *Jessica Kaiser* /

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

Dated: December 17, 2025