

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner

v.

KOVE IO, INC.,
Patent Owner

Patent No. 7,233,978

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 7,233,978**

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	MANDATORY NOTICES	1
III.	PAYMENT OF FEES UNDER 37 C.F.R. §§ 42.15 AND 42.103	4
IV.	GROUND FOR STANDING.....	4
V.	PRECISE RELIEF REQUESTED AND GROUNDS RAISED.....	4
VI.	LEVEL OF ORDINARY SKILL	6
VII.	THE '978 PATENT.....	6
VIII.	CLAIM CONSTRUCTION	9
IX.	DETAILED EXPLANATION OF GROUNDS.....	11
A.	Ground 1: Kahn Renders Claims 1, 3, 5, 6, 10, and 12-16 Obvious.....	11
1.	Claim 1	11
2.	Claim 3	24
3.	Claim 5	27
4.	Claim 6	28
5.	Claim 10	29
6.	Claim 12	33
7.	Claim 13	33
8.	Claim 14	34
9.	Claim 15	39
10.	Claim 16	39
B.	Ground 2: Kahn in View of Krasner Renders Claims 2 and 11 Obvious.....	40
1.	Claim 2	40
2.	Claim 11	42
C.	Ground 3: Kahn in View of Vingralek Renders Claims 3-4, 7, 10, 12-17, 23, and 30 Obvious	43

Petition for *Inter Partes* Review
Patent No. 7,233,978

1.	Claim 3	43
2.	Claim 4	48
3.	Claim 7	50
4.	Claim 10	51
5.	Claim 12	52
6.	Claim 13	52
7.	Claim 14	52
8.	Claim 15	53
9.	Claim 16	54
10.	Claim 17	55
11.	Claim 23	63
12.	Claim 30	64
D.	Ground 4: Kahn in View of Vingralek and Krasner Renders Claim 11 Obvious.....	64
	1. Claim 11	64
E.	Ground 5: Kahn in View of Vingralek and Sato Renders Claims 24-26 Obvious	65
	1. Claim 24	65
	2. Claim 25	68
	3. Claim 26	69
X.	DISCRETIONARY DENIAL IS NOT APPROPRIATE	71
A.	The Board Should Not Deny Institution Under § 325	71
B.	Discretionary Denial Under <i>General Plastic</i> Is Not Appropriate.....	72
C.	The Board Should Not Deny Institution Under <i>Fintiv</i>	73
XI.	CONCLUSION.....	75
XII.	APPENDIX A (CLAIM LISTING)	76
	Claim 1	76
	Claim 2	77
	Claim 3	77

Petition for *Inter Partes* Review
Patent No. 7,233,978

Claim 4	78
Claim 5	78
Claim 6	78
Claim 7	78
Claim 10	79
Claim 11	80
Claim 12	80
Claim 13	80
Claim 14	80
Claim 15	81
Claim 16	82
Claim 17	82
Claim 23	83
Claim 24	83
Claim 25	84
Claim 26	84
Claim 30	84

LIST OF EXHIBITS

EX1001	U.S. Patent No. 7,233,978 to Overton et al. (“the ’978 patent”)
EX1002	Declaration of Sandeep Chatterjee, Ph.D.
EX1003	Curriculum Vitae of Sandeep Chatterjee, Ph.D.
EX1004	Prosecution History of U.S. Patent No. 7,233,978
EX1005	WIPO Application No. 1997/043717 to Kahn et al.
EX1006	U.S. Patent No. 6,473,781 to Skagerwall et al.
EX1007	R. Vingralek, <i>Distributed file organization with scalable cost/performance</i> (1993)
EX1008	RESERVED
EX1009	U.S. Patent No. 6,104,338 to Krasner
EX1010	RESERVED
EX1011	Japanese Patent Application Publication No. H09-231180 to Sato et al.
EX1012	RESERVED
EX1013	U.S. Patent No. 6,594,692
EX1014	RFC 768, User Datagram Protocol (Aug. 28, 1980)
EX1015	M. C. Norrie et al., <i>The Database Research Group at ETH Zurich</i> , SIGMOD Record (March 1994)
EX1016	U.S. Patent No. 5,764,756
EX1017- EX1018	RESERVED
EX1019	First Amended Complaint (Dkt. No. 36) in <i>Kove IO, Inc. v. Google LLC</i> , Case No. 1:23-cv-04244 (N.D. Ill. Nov. 14, 2023)

Petition for *Inter Partes* Review
Patent No. 7,233,978

EX1020	Declaration of Sylvia D. Hall-Ellis, Ph.D.
EX1021	RESERVED
EX1022	<i>Amazon Web Services, Inc. v. Kove IO, Inc.</i> , IPR2020-00276, Paper 23 (June 16, 2020)
EX1023	Kove’s Memorandum in Opposition to Amazon Web Services, Inc.’s Motion for Judgment on the Pleadings (Dkt. Nos. 145 and 145-1) in <i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , Case No. 1:18-cv-08175 (N.D. Ill. May 29, 2020)
EX1024	Memorandum Opinion and Order (Dkt. No. 484) in <i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , Case No. 1:18-cv-08175 (N.D. Ill. Dec. 17, 2021)
EX1025	Memorandum Opinion and Order (Dkt. No. 739) in <i>Kove IO, Inc. v. Amazon Web Services, Inc.</i> , Case No. 1:18-cv-08175 (N.D. Ill. Feb. 6, 2024)
EX1026- EX1032	RESERVED
EX1033	2020 to 2023 Northern District of Illinois Orders Regarding Motions to Stay
EX1034	Agreed Order Setting Amended Patent Case Schedule (Dkt. No. 49) in <i>Kove IO, Inc. v. Google LLC</i> , Case No. 1:23-cv-04244 (N.D. Ill. Feb. 20, 2024)
EX1035	Federal Court Management Statistics (December 2023)
EX1036	Memorandum by Director Vidal (June 21, 2022)
EX1037- EX1041	RESERVED
EX1042	<i>Amazon Web Services, Inc. v. Kove IO, Inc.</i> , IPR2020-00276, Paper 2 (Dec. 12, 2019)

Petition for *Inter Partes* Review
Patent No. 7,233,978

EX1043	Request for <i>Ex Parte</i> Reexamination of U.S. Patent No. 7,233,978 in <i>Ex Parte</i> Reexamination No. 90/019,034 (Nov. 19, 2021)
EX1044	Order Granting Request for <i>Ex Parte</i> Reexamination in <i>Ex Parte</i> Reexamination No. 90/019,034 (Dec. 23, 2021)
EX1045	Notice of Intent to Issue <i>Ex Parte</i> Reexamination Certificate in <i>Ex Parte</i> Reexamination No. 90/019,034 (June 28, 2023)
EX1046	Notice of Termination of <i>Ex Parte</i> Reexamination No. 90/019,108 (Oct. 27, 2022)
EX1047	Request for <i>Ex Parte</i> Reexamination of U.S. Patent No. 7,233,978 in <i>Ex Parte</i> Reexamination No. 90/019,109 (Aug. 31, 2022)
EX1048	Order Granting Request for <i>Ex Parte</i> Reexamination in <i>Ex Parte</i> Reexamination No. 90/019,109 (Nov. 22, 2022)
EX1049	Notice of Intent to Issue <i>Ex Parte</i> Reexamination Certificate in <i>Ex Parte</i> Reexamination No. 90/019,109 (Nov. 28, 2023)
EX1050	Request for <i>Ex Parte</i> Reexamination of U.S. Patent No. 7,233,978 in <i>Ex Parte</i> Reexamination No. 90/019,162 (Feb. 17, 2023)
EX1051	Order Granting Request for <i>Ex Parte</i> Reexamination in <i>Ex Parte</i> Reexamination No. 90/019,162 (May 16, 2023)
EX1052	Order for Staying Discovery Process until July 15, 2024 (Dkt. No. 65) in <i>Kove IO, Inc. v. Google LLC</i> , Case No. 1:23-cv-04244 (N.D. Ill. May 1, 2024)
EX1053	Northern District of Illinois Local Patent Rules
EX1054	Petition Decision on Request for Reconsideration in <i>Ex Parte</i> Reexamination No. 90/019,162 (June 5, 2024)
EX1055	Google’s Motion to Stay and for Expedited Briefing Schedule (Dkt. No. 64) in <i>Kove IO, Inc. v. Google LLC</i> , Case No. 1:23-cv-04244 (N.D. Ill. Apr. 30, 2024)
EX1056	Joseph B. Greene, <i>Oracle8 Server Unleashed</i> (1998)

EX1057	Joseph B. Greene, <i>Oracle DBA Survival Guide</i> (1995)
EX1058	Betty Salzberg, <i>File Structures</i> (1988)
EX1059	Andrew P. Black & Yeshayahu Artsy, <i>Implementing Location Independent Invocation</i> (1989)

I. INTRODUCTION

Google LLC (“Petitioner”) requests *inter partes* review of claims 2-7, 10-17, 23-26, and 30 (“the challenged claims”) of U.S. Patent No. 7,233,978 (“the ’978 patent”) (EX1001) assigned to Kove IO, Inc. (“Patent Owner” or “PO”).

II. MANDATORY NOTICES

Real Parties-in-Interest: Pursuant to 37 C.F.R. § 42.8(b)(1), Petitioner identifies Google LLC as the real party-in-interest.¹

Related Matters: The ’978 patent is asserted in *Kove IO, Inc. v. Amazon Web Services, Inc.*, Case No. 1:18-cv-08175 (N.D. Ill.) (“the AWS Proceeding”), and *Kove IO, Inc. v. Google LLC*, Case No. 1:23-cv-04244 (N.D. Ill.).

The ’978 patent was at issue in *Amazon Web Services, Inc. v. Kove IO, Inc.*, IPR2020-00276 (“the AWS IPR”). Amazon Web Services (“AWS”) filed the AWS IPR (EX1042) in December 2019 presenting grounds based on different combinations of prior art from the grounds here (*id.*, 23-24; *infra* §§IV, IX), and institution was denied (EX1022, 24). Petitioner had no involvement in the preparation or filing of the AWS IPR.

¹ Google LLC is a subsidiary of XXVI Holdings Inc., which is a subsidiary of Alphabet Inc. XXVI Holdings Inc. and Alphabet Inc. are not real parties-in-interest to this proceeding.

The '978 patent was also at issue in ex parte reexamination Nos. 90/019,034 (“the '034 EPR”), 90/019,108 (“the '108 EPR”), and 90/019,109 (“the '109 EPR”), and is currently pending in ex parte reexamination No. 90/019,162 (“the '162 EPR”).

The '034 EPR was filed by third party AWS (EX1043) in November 2021 presenting SNQs on claims 1, 3, 6, 10, 14, 17, 23, 24, 30, and 31 based on different combinations of prior art from the grounds here (*id.*, 12-14; *infra* §§IV, IX); the request was granted (EX1044, 18), with claims 1 and 31 being canceled and the rest of the claims confirmed (EX1045, 5). Petitioner had no involvement in the preparation or filing of the '034 EPR.

The '108 EPR was filed by third party AWS in August 2022 but was terminated for failing to comply with filing requirements. (EX1046, 3.) Petitioner had no involvement in the preparation or filing of the '108 EPR.

The '109 EPR was filed by third party AWS (EX1047) in August 2022, presenting SNQs on claims 17, 23, 24, and 30 based on different combinations of prior art from the grounds here (*id.*, 15-18; *infra* §§IV, IX). The request was granted with respect to all of claims 17-30, with the examiner including claims 18-22 and 25-29 *sua sponte* because they all depend from claim 17. (EX1048, 39-40.) The claims were confirmed. (EX1049, 8.) Petitioner had no involvement in the preparation or filing of the '109 EPR.

The '162 EPR was filed by third party AWS (EX1050) in February 2023, presenting SNQs on claims 3, 6, 10, and 14 based on different combinations of prior art from the grounds here (*id.*, 21-24; *infra* §§IV, IX.). Petitioner had no involvement in the preparation or filing of the '162 EPR. Because the request raised substantial new questions of patentability, reexamination was ordered on claim 6 in May 2023 (EX1051, 52), and then on all remaining claims 3, 10, and 14 in June 2024 after a decision on AWS's petition for review (EX1054, 9). The examiner has not issued an Office Action at the time of the filing of the current Petition.

Petitioner is concurrently filing IPR petitions challenging related U.S. Patent Nos. 7,103,640 ("the '640 patent") and 7,814,170 ("the '170 patent"), which are also at issue in the above-noted district court cases.

Counsel and Service Information: Lead counsel is Naveen Modi (Reg. No. 46,224). Backup counsel are Joseph E. Palys (Reg. No. 46,508), Daniel Zeilberger (Reg. No. 65,349), Howard Herr (*pro hac vice* to be filed), Alexa Lowman (*pro hac vice* to be filed), and Puja Patel (*pro hac vice* to be filed). Service information is Paul Hastings LLP, 2050 M Street NW, Washington, D.C. 20036, Tel.: 202.551.1700, Fax: 202.551.1705, email: PH-Google-Kove-IPR@paulhastings.com. Petitioner consents to electronic service.

III. PAYMENT OF FEES UNDER 37 C.F.R. §§ 42.15 AND 42.103

The PTO is authorized to charge any fees due during this proceeding to Deposit Account No. 50-2613.

IV. GROUNDS FOR STANDING

Petitioner certifies that the '978 patent is available for review and Petitioner is not barred/estopped from requesting review on the identified grounds identified.

V. PRECISE RELIEF REQUESTED AND GROUNDS RAISED

Claim 1 has been canceled. Petitioner addresses claim 1 in Ground 1 for its dependent claims 2-7. The challenged claims should be canceled as unpatentable on the following grounds under pre-AIA 35 U.S.C. § 103:

Ground	Claim(s)	References
1	1, 3, 5, 6, 10, 12-16	Kahn (EX1005)
2	2, 11	Kahn and Krasner (EX1009)
3	3-4, 7, 10, 12-17, 23, 30	Kahn and Vingralek (EX1007)
4	11	Kahn, Vingralek, and Krasner
5	24-26	Kahn, Vingralek, and Sato (EX1011)

The earliest possible effective filing date for the '978 patent is July 8, 1998.²

The asserted prior art herein qualifies as prior art as follows:

Kahn (filed: 05/09/1997; published: 11/20/1997)	§§102(a), (b)
Sato (published: 09/05/1997)	
Vingralek (published: 12/1993)	§102(b)
Krasner (filed: 05/14/1998; issued: 08/15/2000)	§102(e)

The declaration of Dr. Sylvia D. Hall-Ellis (Ex. 1020, ¶¶1-66) confirms that Vingralek was published, disseminated, and reasonably available to the public on or shortly after December 31, 1993. (Ex. 1020, ¶¶46-54; *see also id.*, ¶¶19-45.) Dr. Hall-Ellis explains that based on finding a copy of Vingralek that is dated December 1993 in the ETH Zürich Library and a MARC record in its online library catalog indicating that the ETH Zurich journal was first published in 1974. (*Id.*, ¶¶51-54.) Moreover, Vingralek was cited by Ex. 1015 (“Norrie”), which was published on March 1, 1994. (*Id.*, ¶¶55-63.) It is her opinion that Vingralek was

² Petitioner believes the '978 patent is not entitled to a priority date earlier than August 13, 1999, but does not raise that issue in this petition as it is not relevant. Petitioner does not concede that the '978 patent is entitled to either the July 8, 1998 or August 13, 1999 priority date.

publicly available on or shortly after December 31, 1993, as shown on the cover.

(*Id.*, ¶¶64-65.)

VI. LEVEL OF ORDINARY SKILL

A person of ordinary skill in the art at the time of the alleged invention of the '978 patent ("POSITA") would have had an undergraduate degree in electrical engineering, computer engineering, computer science or a related field along with two years of work experience in the field of distributed computing systems.

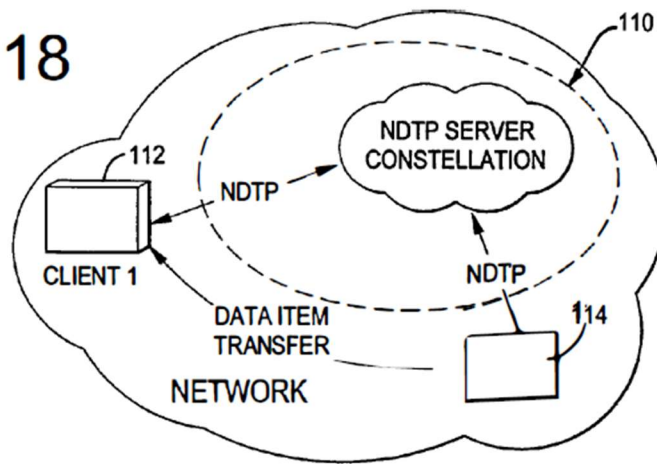
(EX1002, ¶¶20-21.)³ More education can supplement practical experience and vice versa. (*Id.*)

VII. THE '978 PATENT

The '978 patent describes storing/retrieving information across a network (EX1001, 1:30-33, 1:56-58) and a network-distributed-tracking protocol (NDTP) (*id.*, 2:19-21, 7:55-61). Figure 18 shows an NDTP server constellation where servers return strings in response to client requests. (*Id.*, 18:50-58.)

³ Petitioner submits the declaration of Dr. Sandeep Chatterjee (EX1002), an expert in the field of the '978 patent. (EX1002, ¶¶1-28; EX1003.)

FIG. 18



(*Id.*, FIG. 18; EX1002, ¶29.)

Server constellations have two organizational approaches: client-centric (Figure 19) and server-centric (Figure 20). In the client-centric approach, processing emphasis is toward clients (*id.*, 18:60-19:3), whereas in the server-centric approach, server-processing is emphasized (*id.*, 19:20-24).

FIG. 19

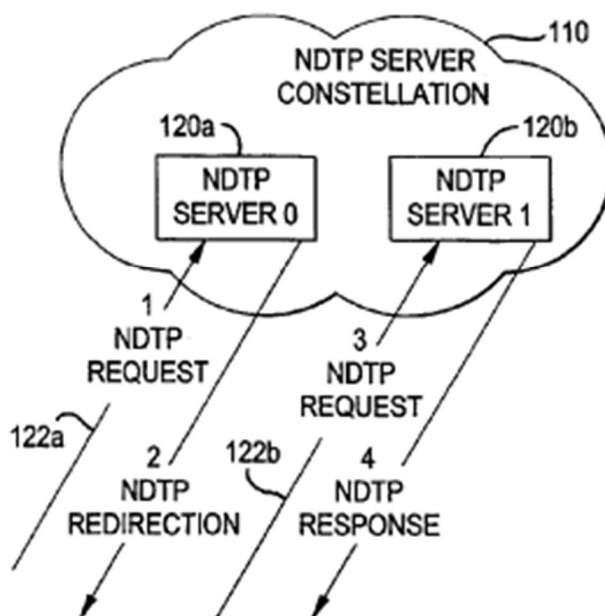
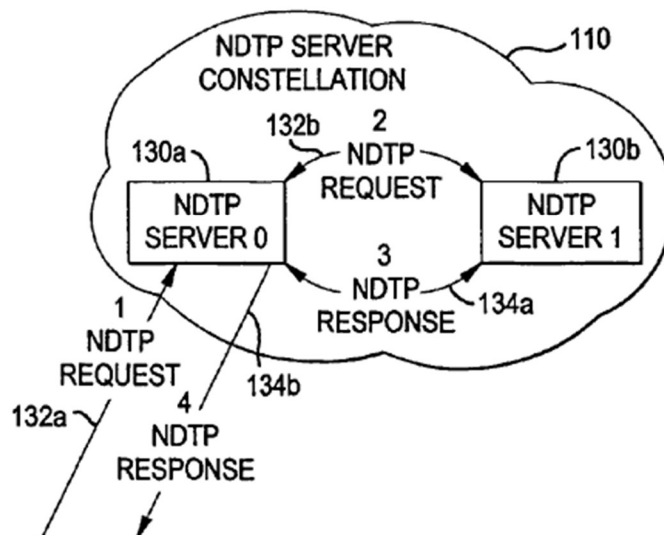


FIG. 20



(*Id.*, FIGS. 19, 20; EX1002, ¶30.)

In both approaches, a client transmits an identifier specifying an entity to a server in a client request. (EX1001, 2:19-52.) If the server contains location information specifying network locations of data associated with the entity, the server returns the location information. (*Id.*; EX1002, ¶31.)

As demonstrated below, all of the features recited in the challenged claims of the '978 patent were known in the prior art before the alleged invention date of the '978 patent. (*Infra* §IX; EX1002, ¶¶22-28, 32-44; EX1006, EX1011, EX1056-EX1059.)

VIII. CLAIM CONSTRUCTION

The Board only construes the claims when necessary to resolve the underlying controversy. *Toyota Motor Corp. v. Cellport Sys., Inc.*, IPR2015-00633, Paper No. 11 at 16 (Aug. 14, 2015). For purposes of this proceeding, Petitioner believes that no special constructions are necessary to assess whether the challenged claims are unpatentable over the asserted prior art. (EX1002, ¶¶45-51.)⁴

As shown below, in the AWS IPR, the Board construed a claim term of the '978 patent (EX1022, 13), and in the AWS Proceeding, the district court construed

⁴ Petitioner reserves all rights to raise claim construction and other arguments in other venues.

several claim terms of the '978 patent. (EX1024, 35; EX1025, 19.) While Petitioner does not acquiesce to those constructions, the prior art mappings in §IX explain how the claims of the '978 patent are disclosed and/or rendered obvious if those constructions are applied. (EX1002, ¶¶45-51.)

'978 Patent Terms	AWS Proceeding/IPR Construction
"identifier"	"a unique encoding that identifies an individual entity, and with which zero or more data location specifiers are associated in a location server" (EX1022, 13.)
"location information"	"one or more identifiers and their associated locations" (EX1024, 35.)
"location server"	"a network-attached component that maintains a set of identifier/location mappings that are modified or returned in response to location request messages from clients" (EX1024, 35.)

'978 Patent Terms	AWS Proceeding/IPR Construction
“client”	“a network-attached component (which may be software or hardware) that initiates update or lookup of identifier/location mappings from a location server with location request messages” (EX1024, 35.)
“plurality of location servers”	“location servers in non-hierarchical structures” (EX1025, 19.)

IX. DETAILED EXPLANATION OF GROUNDS

A. Ground 1: Kahn Renders Claims 1, 3, 5, 6, 10, and 12-16 Obvious

1. Claim 1

i) [1.pre]

To the extent limiting, Kahn discloses and/or suggests the limitations therein. (EX1002, ¶¶53-62.) Kahn discloses with reference to Figure 1 a system for managing digital objects (green) stored in repositories (blue), where each digital object is identified by a handle (yellow) and located by pointer(s) associated with the handle. (EX1005, FIG. 1, Abstract, 1:10-12, 6:9-10, 6:24, 11:16-18, 21:3-

5.) A digital object may include digital representations of works (e.g., books, software). (*Id.*, 1:16-20.) Kahn discloses that the system includes handle servers (“plurality of location servers”) each managing one or more handles and their associated pointers (collectively, “location information”). For example, each handle server includes a table storing a handle-pointer mapping, and the server accepts handles of the digital objects and returns corresponding pointers indicating the digital objects’ network locations. (*Id.*, Abstract, 6:28-35, 7:2-20, 21:6-24, 37:31-33; EX1002, ¶53.)

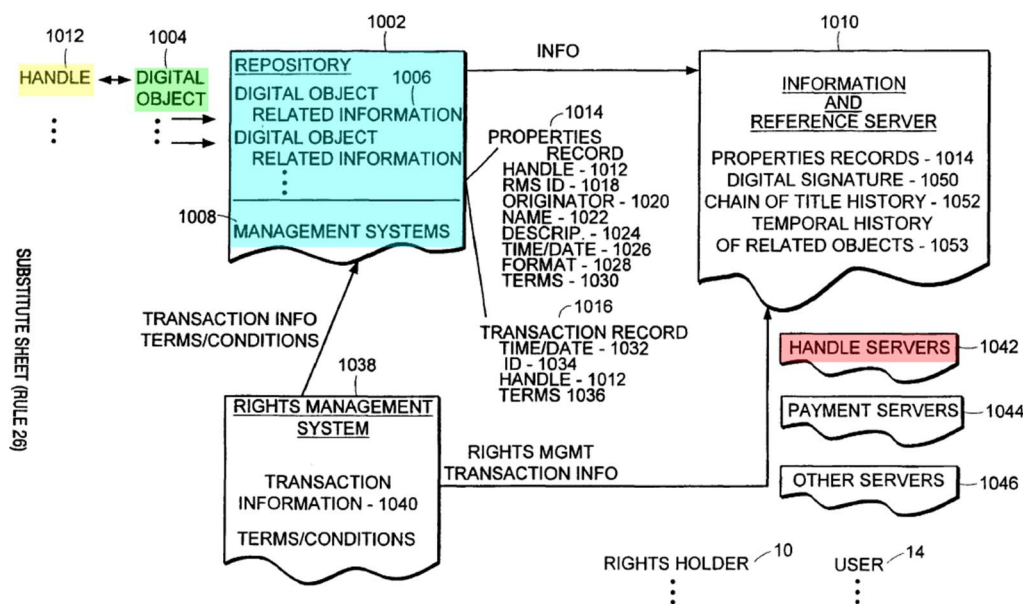


FIG. 1

(EX1005, FIG. 1.)

Kahn discloses that the handle servers (“a plurality of location servers”) provide handles and associated pointers in response to a client’s request to retrieve locations of digital objects (“providing location information to location queries”).

(*Id.*, Abstract.) Figure 8 provides an example of such a process. (*Id.*, FIG. 8, 46:26-28, 46:29-47:25, 72:32-74:32.)

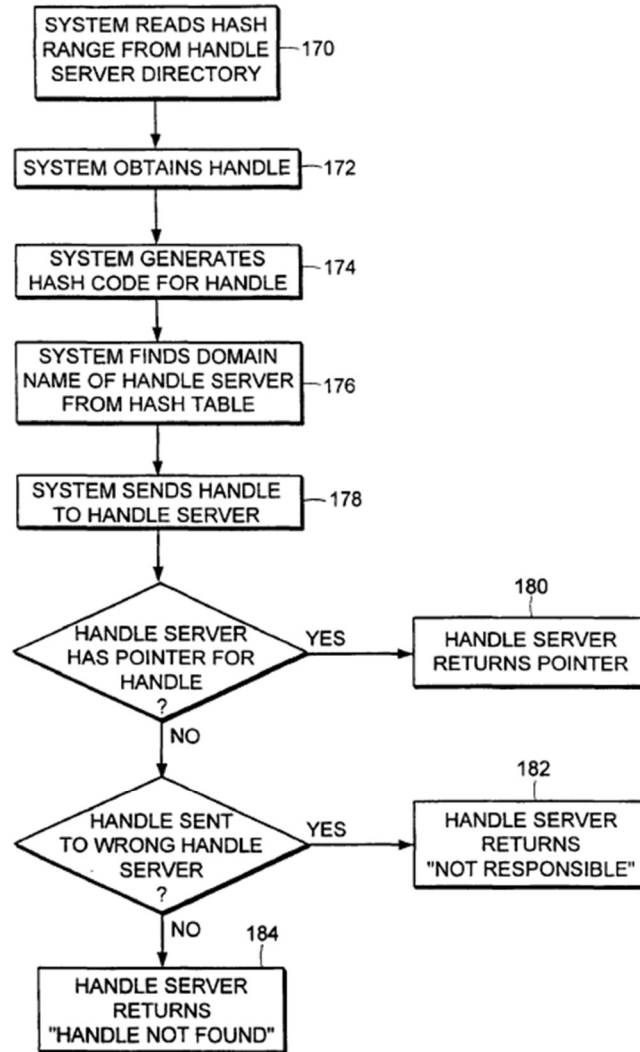


FIG. 8
SUBSTITUTE SHEET (RULE 26)

(*Id.*, FIG. 8.) As discussed for step 170 of Figure 8, a client downloads from a handle directory server a hash range/handle-server-domain-name table (Figure 7 below) that associates hash ranges with handle server domain names. The table allows the client to determine the location of a handle server containing a pointer

for retrieving the digital object associated with the handle. (*Id.*, 29:18-26, 38:9-17, 46:22-47:5, 73:4-74:26.)

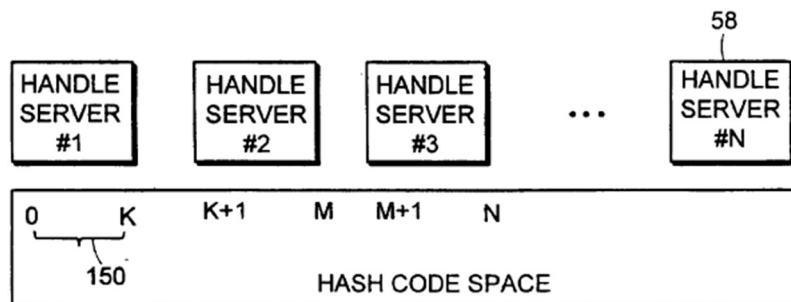


FIG. 7

(*Id.*, FIG. 7.) As shown in Figure 7 above, each handle server is assigned a range of hash codes/values 150. (EX1005, 46:22-24, FIG. 7.) The client generates a hash code for a handle using a predetermined hashing algorithm and then uses the hash code/value as an index into the downloaded table, allowing the client to obtain the domain name or location of the handle server that potentially stores the pointers for locating the digital object associated with the handle. (*Id.*, 47:1-11, 73:17-26; EX1002, ¶54.)

Returning to Figure 8, the client in step 172 receives a handle from a user's request for obtaining a certain digital object. (EX1005, 46:34-47:1, 73:8-16.) The client then "generates the hash code for the handle using a predetermined hashing algorithm," then uses the hash code to index into the table "to determine the domain name of the appropriate handle server," and sends the handle to the handle

server in a request to retrieve the location of the digital object (“location quer[y]”) in steps 174, 176, and 178. (*Id.*, 47:1-8, 73:17-22; EX1002, ¶55.)

The handle server (“location server”) verifies if it does or, in the event of a dynamic change to the table downloaded to the client, does not contain the handle and its associated pointer. (EX1005, 47:10-11, 73:32-74:12.) If the handle is located in the handle server, one or more pointers associated with the handle for locating the digital object are returned to the client in step 180. (*Id.*, 29:8-10, 47:10-13, 73:23-26, 74:21-26; EX1002, ¶56.) The client then uses the pointer(s) to obtain the digital object stored in a repository. (EX1005, 74:21-29.)

Kahn discloses the claimed “location server[]” under the AWS Proceeding’s construction. (§VIII; EX1001, 4:34-38.) Kahn discloses that each handle server (“location server”) is a component attached to a distributed network (i.e., a network-attached component). (EX1005, 7:5-7, 73:27-31.) As discussed for limitation [1.a], each handle is an “identifier” as construed in the AWS IPR. (§IX.A.1.ii; EX1005, 6:27-28, 9:23-26, 11:8-13, 21:3-5.) Each handle server maintains mappings between handles (i.e., identifiers) and pointers (i.e., locations). (EX1005, 7:17-20, 21:6-10, 37:31-33.) And as discussed above, Kahn’s pointers, which identify a digital object’s location, disclose “location[s]” (or similar terms) as described in the ’978 patent. (EX1001, 4:28-30; EX1002, ¶57.)

Further, Kahn discloses that the handle server returns identifier/location mappings in response to a request from the client for location information associated with a handle (i.e., in response to location request messages from a client). For example, as discussed above, pointers (i.e., locations) are returned to the client if the handle server determines it contains the requested location information associated with the handle in the request. Given that the returned pointers are associated with the handle in the location information request, a POSITA would have understood that the returned pointers correspond to mappings of the handle to the pointers, consistent with the '978 patent's disclosure. (EX1001, 2:32-37, 4:61-64.) Moreover, Kahn discloses that the full handle record (i.e., including the handle and all corresponding data) may be returned in response to a client's request, such that both pointers and the handle itself are returned. (EX1005, 42:1-11, 44:16-21; EX1002, ¶58.) Kahn also discloses that identifier/location mappings may be updated/modified in response to a request from the client to add new location information (i.e., another type of location request message from the client). (EX1005, 43:31-44:4, 45:22-46:24; EX1001, 10:1-4.)

Kahn discloses the claimed "plurality of location servers" under the AWS Proceeding's construction. (§VIII.) A POSITA would have understood that the handle servers ("plurality of location servers") are arranged in a flat, cluster

topology, rather than a hierarchical or tree-like structure where servers can only communicate with nodes immediately above or below it in the tree. (EX1002, ¶59; EX1005, 4:19-22, 7:8-9.) Kahn discloses that the hash function is used to distribute handles and associated pointers among the handle servers. (EX1005, 4:19-22, 7:8-9, 37:31-38:3, FIG. 7.) A POSITA would have thus understood that because a hash function is used for distribution across handle servers, Kahn's handle servers are arranged in a non-hierarchical configuration. Indeed, consistent with the above, Patent Owner argued in the AWS Proceeding that the hash function feature requires a non-hierarchical server structure. (EX1025, 15.)

Kahn also discloses the claimed "location information" under the AWS Proceeding's construction. (§VIII.) As discussed above, each handle server includes a table storing a handle-pointer mapping, where the handles are identifiers of digital objects and the associated pointer(s) indicate the location of a stored digital object. Also discussed above, the handle servers provide one or more handles and associated pointers (collectively "location information") in response to a client's request to retrieve locations of digital objects ("providing location information to location queries"). (EX1002, ¶60.)

To the extent PO argues or the Board finds that Kahn does not disclose a handle server returning/providing *both pointers and their associated handle* in response to a location query, as required by the claimed "providing location

information to location queries” under the AWS Proceeding construction of “location information,” a POSITA would have found it obvious to include such a feature in Kahn. (EX1002, ¶61.) A POSITA would have found it beneficial to implement this feature because returning the handle with its associated pointers would allow the client to verify the returned pointers are indeed associated with the same handle in the request. Indeed, Kahn discloses a similar process to verify “whether a purported instance of a digital object is identical to the original” by using a “digital object identifier” called a handle. (EX1005, 3:26-30; *see also id.*, 1:10-12.) Moreover, this is particularly beneficial when the client transmits multiple requests for pointers, and the feature (i.e., returning handle with the requested pointers) would have allowed the client to conveniently track the request status and determine which pointers are for which requests based on the returned handles, consistent with Kahn’s goal of “insur[ing] the integrity of the transactions within the system.” (*Id.*, 26:16-27.) Additionally, Kahn consistently discloses including a handle in the process of retrieving a digital object, e.g., a user’s request to a client for a digital object includes a handle associated with the object, and the request from client to handle server includes the same handle. (*Id.*, 16:6-8, 46:34-47:8, 73:8-22.) Thus, a POSITA would have found it obvious to include the claimed feature. (EX1002, ¶61.)

A POSITA would have also had a reasonable expectation of success in implementing this modification. Kahn discloses that each handle server includes a table that associates handles with pointers. (EX1005, 7:17-20, 37:31-33.) Thus, the handle server would be readily able to transmit the handle with the associated pointers to the client. Moreover, Kahn is already capable of transmitting information along with its associated handle when the information is accessed (*id.*, 23:12-19, 23:33-34) and thus a POSITA would have understood that pointers may likewise be transmitted with their associated handle(s). Such a modification would have involved applications of known technologies (e.g., known handle server storing includes a table that associates handles with pointers) according to known methods (e.g., known handle server providing requested location pointers based on a client's request using a handle) to yield the predictable result of a handle server returning both a handle and associated pointers in response to a client's request, as discussed above. *KSR Int'l. Co. v. Teleflex Inc.*, 550 U.S. 398, 416 (2007). Accordingly, Kahn discloses and/or suggests this limitation. (EX1002, ¶62.)

ii) [1.a]

Kahn discloses this limitation. (EX1002, ¶¶63-66.) **First**, Kahn explains the system contains multiple handle servers, including a handle server (“first location server”) that contains one or more handles (“identifier”) associated with one or more pointers (“at least one location string associated with the identifier”)

(handle and pointer collectively the “first set of location information...the location information comprising an identifier and at least one location string”), where the handle and pointer collectively indicate an object’s location (“first set of location information corresponding to at least one entity”). In particular, each of the multiple handle servers in a network contains a set of handle-to-pointer mappings, where each mapping consists of a handle associated with one or more pointers. (EX1005, 7:17-20, 21:6-10, 28:31-34, 37:31-33.) While Kahn discloses that a handle may be associated with zero pointers when an object is a physical object, a POSITA would have understood that, when a system includes all digital objects, the system would have at least one pointer (“at least one location string”) associated with each handle (“identifier”). (*Id.*, 29:8-12, 37:23-24.) Further, Kahn discloses that one or more handles and associated pointers (collectively, “location information”) correspond to at least one object (“at least one entity”), as each handle identifies an object and associated pointers indicate the object’s location in the system. (*Id.*, 9:23-26, 21:3-5, 28:31-34; EX1002, ¶63.)

Second, Kahn discloses that each digital object is associated with a unique handle (“identifier”) identifying that object (“entity”), where the handle comprises a unique character string. (EX1005, 7:4-5.) In particular, Kahn explains that “[e]ach digital object has a ‘handle,’ a concise unique identifier for a digital object

used for storage and retrieval operations and other repository functions.”

(EX1005, 21:3-5; *see also id.*, 6:27-28, 9:23-26, 11:8-13; EX1002, ¶64.)

Kahn further discloses pointers (“location string[s]”) indicate the location of sequences of bits/digits (“data”) pertaining to an object (“specifies a location of data pertaining to the entity”) in the network. (EX1002, ¶65.) Kahn discloses that each object (“entity”) is represented by sequences of bits/digits, which a POSITA would have understood are “data.” (EX1005, 1:10-12, 9:23-26, 11:16-18.) Indeed, Kahn discloses that “a digital object could include programs and data” which enable delivery over a network or reproduction on a computer screen. (*Id.*, 1:20-25.) Additionally, Kahn discloses that a pointer indicates the location of “data” pertaining to an object in the network. (*Id.*, 1:10-12, 4:25-29, 6:30-35, 28:31-32, 29:8-12.)

Kahn’s handle is an “identifier” under the AWS IPR construction. (§VIII.) As discussed above, Kahn discloses that a handle is a unique character string (i.e., a unique encoding) that identifies a digital object (i.e., an individual entity). (EX1002, ¶66.) A POSITA would have understood that the handle is a unique encoding because each digital object is uniquely represented and encoded in the system by the characters that comprise the handle. (EX1005, 33:3-14, 33:22-27.) As discussed above, zero or more pointers (i.e., data location specifiers), which specify the location of a digital object (including “data”), are associated with a

handle in a handle server (i.e., a location server, as construed in the AWS Proceeding). (§IX.A.1.i.)

iii) [1.b]

Kahn discloses this limitation. (EX1002, ¶¶67-68.) **First**, as discussed for limitations [1.pre]-[1.a], Kahn discloses multiple handle servers in a network, where each server contains a set of handle-to-pointer mappings, and each mapping consists of a handle associated with one or more pointers. (§§IX.A.1.i-ii.) Thus, in addition to the “first location server containing a first set of location information,” Kahn discloses another handle server (“second handle server”) contains another set of handles and associated pointers (“second set of location information”). (§§IX.A.1.i-ii; EX1005, 4:25-29, 6:30-35, 7:4-20, 28:31-32; EX1002, ¶67.)

Second, Kahn explains that each handle is unique to a digital object. (EX1005, 21:3-5.) Further, the second handle server stores a handle and pointer for a second digital object. (*Id.*, 4:25-29, 37:31-38:3.) Thus, a second handle server contains one or more unique handles associated with pointers which differ from a first handle server with its own unique handles associated with pointers (“a portion of the second set of location information differs from the first set of location information”). (EX1002, ¶68.)

iv) [1.c]

Kahn discloses this limitation. (EX1002, ¶¶69-71.) **First**, Kahn discloses each of the multiple handle servers (“each of the location servers”) may be implemented using “hardware and software” and “can be implemented as one or more server computers.” (EX1005, 26:28-31, 42:2-3; *see also id.*, 21:6-10, 29:5-12, 44:6-7.) The handle system also interfaces with handle servers using routines written in the C programming language. (*Id.*, 40:31-41:2.) As such, a POSITA would have understood that each handle server performs associated processes using certain configured “programming logic” stored on each handle server. (EX1002, ¶¶69.)

Second, Kahn discloses that a handle server (having “programming logic” stored and configured thereon) responds to a client’s request (“responsive to a location query”) identifying a desired object (“identifying a desired entity”) by returning a message containing a list of pointers (“return a location message”) indicating the digital object’s location. A user requests a search for a digital object by using an object’s handle as input to the client, (*id.*, 73:8-14), and the client then performs processes (e.g., identifying a correct handle server to send the request) to locate the digital object associated with the handle. (§IX.A.1.i; EX1005, 47:1-7, 73:17-26.) A handle query (“location query”) containing the handle identifying a desired object (“identifying a desired identity”) is sent via request-for-pointer-

information UDP packet from the client to the correct handle server (having “programming logic” stored and configured thereon). (§IX.A.1.i; EX1005, 7:5-7, 29:5-12, 43:20-23, 44:6-7, 46:34-47:11, 73:8-34.) “If the handle server found the pointers, a list of pointers is returned....” (EX1005, 47:9-11, 73:21-27, FIG. 8 (step 180); §IX.A.1.i; EX1002, ¶70.)

Third, as discussed above, Kahn discloses returning a message containing a list of pointers (“location message”) indicating the location of a digital object, i.e., the sequence of bits/digits (“location of data”) pertaining to the requested object (“desired entity”). This message is sent if the handle server receiving the client’s request (“location server receiving the location query”) contains the handle and associated pointers for the requested object (“location information for the desired entity”). If the handle server is responsible for the handle associated with the requested object, one or more associated pointers stored on the handle server for locating the object are returned for use by the client. (EX1005, 47:10-11, 73:21-74:26; EX1002, ¶71.)

2. Claim 3

Kahn discloses claim 3. (EX1002, ¶¶72-76.) **First**, as discussed for limitation [1.c], Kahn discloses the handle server (having “programming logic” stored and configured thereon) is responsive to a client’s request to retrieve the location of a digital object (“logic responsive to the location query”). (§IX.A.1.iv.)

Kahn further discloses the handle server responds to this request by returning either a message containing a list of associated pointers (“location message”) or the claimed “redirect message,” described in detail below. (EX1002, ¶72.)

Second, Kahn discloses a handle server, which receives a request (“location server receiving the query”) to retrieve a digital object, returns a message if the handle server contains the handle and associated pointers for the object (“returns the location message if the queried location server contains information for the desired entity”). (EX1005, Abstract, 29:8-12, 47:9-10, 73:32-74:26.) In particular, the returned message contains a list of pointers (“location message”) indicating the locations of an object. (§IX.A.1.iv; EX1002, ¶73.)

Third, Kahn discloses “wherein the queried location server returns a redirect message if the queried location server lacks location information for the desired entity, the redirect message comprising information for finding a location server known to have location information relevant to the location query.” In the present district court litigation, PO reads the claimed “redirect message” on a cache miss

response due to stale/incorrect location information.⁵ (EX1019, ¶¶88-89.) Under PO's interpretation, Kahn discloses this limitation. (EX1002, ¶74.)

Kahn discloses that the queried handle server ("queried location server") transmits a not-responsible-for-handle message ("returns a redirect message") to the client in the form of an invalid-handle-server-selected response UDP packet if the handle server, i.e., a "wrong handle server," does not contain the handle and its associated pointers (collectively, "location information") for the requested object ("if the queried location server lacks location information for the desired entity"). (EX1005, 47:13-15, FIG. 8 (step 182); *see also id.*, 73:1-74:26, 73:32-74:4, FIG. 25; EX1002, ¶75.) The not-responsible-for-handle message ("redirect message") informs the client to download an updated table from the handle server directory and recalculate the location of a correct handle server containing the handle and associated pointers (collectively, "location information") associated with the request ("comprising information for finding a location server known to have location information relevant to the location query"). (*Id.*, 47:13-19, 74:5-9.)

⁵ Petitioner does not acquiesce to PO's interpretation of the "redirect message." Nevertheless, Petitioner explains how Kahn discloses the "redirect message" under PO's interpretation. Petitioner also explains how Kahn discloses the "redirect message" if the "redirect message" is interpreted differently.

Accordingly, under PO's interpretation, Kahn's not-responsible-for-handle-message discloses the claimed "redirect message" that "compris[es] information for finding a location server known to have location information relevant to the location query" because it provides information which informs the client to download an updated table to determine the location of a correct handle server. (EX1002, ¶76; §IX.C.1 (addressing "redirect message" under other interpretations).)

3. Claim 5

Kahn discloses claim 5. (EX1002, ¶¶77-78.) As discussed for limitations [1.pre]-[1.b], Kahn discloses each of the handle servers ("each of the location servers") stores a portion of handles and associated pointers ("at least a portion of the location information") in the network. (§§IX.A.1.i-iii; EX1005, 4:19-22, 7:2-20, 37:31-38:3.) A POSITA would have understood that in order to maintain handles and associated pointers (collectively, "location information") on each of the handle servers ("location servers"), such information would have necessarily been stored on a persistent storage device. (EX1002, ¶77.) Otherwise, such information for locating/tracking digital objects would have been lost after the handle server powers off. Indeed, Kahn discloses that handle servers store "items that must be retained over very long periods of time" (EX1005, 38:18-28) and servers may "be used to back up each other" (*id.*, 39:7-9). (EX1002, ¶78.)

4. Claim 6

Kahn discloses claim 6. (EX1002, ¶¶79-80.) **First**, Kahn discloses that handles and associated pointers (collectively, “location information”) in a handle server (“location server”) are maintained in a table associating the handles with pointers (“indexed location store”). In particular, Kahn explains that each handle server has a table that associates handles with pointers. (EX1005, 7:17-20, 37:31-33.) A POSITA would have understood that the handle server has a storage medium, e.g., a persistent storage device as discussed for claim 5, to store the table of pointers and associated handles (“location store”). (§IX.A.3.) A POSITA would have further understood that the table in the handle server is an “indexed location store” because each handle serves as an index to retrieve the associated one or more pointers (part of the “location information”). (EX1002, ¶79.)

Second, Kahn discloses a table in the handle server associating sets of handles with one or more pointers (“indexed location store”). Kahn explains that the handles and associated one or more pointers in the table within the handle server are indexed by a hash range/handle-server-domain-name table (“indexed by a hash table”). (EX1005, 47:1-11, 73:17-27.) Specifically, the client “generates the hash code for the handle using a predetermined hashing algorithm,” and then uses the hash code to index into the hash range/handle-server-domain-name table “to determine the domain name of the appropriate handle server” containing the

table within the handle server storing the sets of handles and associated pointers.

(EX1005, 47:1-5; EX1002, ¶80.)

5. Claim 10

i) [10.pre]

Kahn discloses and/or suggests this limitation for the same reasons as discussed for limitation [1.pre]. (§IX.A.1.i; EX1002, ¶81.)

ii) [10.a]

Kahn discloses this limitation. (EX1002, ¶¶82-85.) **First**, as discussed for limitations [1.pre]-[1.a], Kahn explains that multiple handle servers (“plurality of location servers”) contain sets of handles and associated pointers (collectively “location information”) corresponding to multiple objects (“plurality of entities”). (§§IX.A.1.i-ii; EX1002, ¶82.)

Second, Kahn discloses the handle system is “based on the UDP protocol,” which “enables a large number of transactions to be handled efficiently.” (EX1005, 26:8-14, FIG. 33.) Thus, a POSITA would have understood that the sets of handles and associated pointers (collectively, “location information”) are formatted according to the UDP protocol for facilitating transactions (e.g., returning a handle and associated pointers to a client) (“a transfer protocol configured for manipulating location information”). (EX1002, ¶83.)

Third, Kahn discloses multiple handle servers each storing sets of handles and associated pointers (collectively, “location information”). (§§IX.A.1.i-ii.) A pointer contains the domain name of the repository which contains the associated digital object. (EX1005, 29:5-12.) Additionally, Kahn discloses that there are multiple repositories, each repository containing digital objects which users retrieve the location of via client request (i.e., location query). (EX1005, 6:22-24; §§IX.A.1.i, IX.A.1.iv.) Kahn discloses that each repository stores information, performs services, and contains management systems associated with certain tasks, such as those associated with a repository access protocol for adding and accessing digital objects. (EX1005, 11:22-27, 13:2-4, 22:11-23.) A POSITA would have understood that each repository would have implemented a server (i.e., application server) to provide the programming, services, information storage, and management systems required for implementing a repository access protocol. Thus, a POSITA would have understood that the pointers (part of the “location information”) provide an “application server address” because a pointer indicates the domain name of a repository, and a repository has a server with an address. (EX1002, ¶84.)

Fourth, as discussed for limitation [1.pre], a POSITA would have understood that the handle servers (“plurality of location servers”) are arranged in a cluster structure (“arranged in a cluster topology”). (§IX.A.1.i.) Kahn further discloses that the handles and associated pointers are distributed across a plurality

of handle servers, such that each handle server stores handle-to-pointer mappings related to certain digital objects. (EX1005, 29:23-26, 37:31-38:3.) Thus, Kahn discloses that each handle server (“location server”) contains a unique set of handles and associated pointers (collectively, “location information”) out of the full set of all handles and associated pointers (“an aggregate set of the location information”) in the network. (EX1002, ¶85.)

iii) [10.b]

Kahn discloses this limitation. (EX1002, ¶¶86-88.) **First**, Kahn discloses “programming logic stored on each of the plurality of location servers” that is configured to perform any of the actions a “location server” must perform. As discussed for limitation [1.c], Kahn discloses that each of the multiple handle servers (“plurality of location servers”) may be implemented using hardware and software and as a computer, and that the handle system interfaces with handle servers using routines in the C programming language. (§IX.A.1.iv.) Thus, a POSITA would have understood that operations/services provided on the computer are operated using associated “programming logic” stored and configured on each handle server to perform each of the handle server’s operations, including those recited in limitation [10.b] and discussed below. (EX1002, ¶86.)

Second, Kahn discloses “programming logic...responsive to a location query for a desired identifier to return one of...a location message, wherein a

queried location server returns a location message if the queried location server contains location information for the desired identifier.” As discussed for limitation [1.pre], Kahn discloses that the queried handle server (“queried location server”) receives a request (“location query”) from the client for pointers associated with the handle in the request (“for a desired identifier”). (§IX.A.1.i.) As discussed for claim 3, Kahn discloses that the handle server (“queried location server”) returns a message (“location message”) if the queried handle server (“queried location server”) contains the handle and associated pointers (collectively, “location information”) for the desired handle (“identifier”). (§IX.A.2; EX1002, ¶87.)

Third, Kahn discloses “programming logic...responsive to a location query for a desired identifier to return...a redirect message if the queried location server does not contain location information relevant to the desired identifier, wherein the redirect message comprises information for finding a location server having location information related to the desired identifier.” As discussed for limitation [1.pre], Kahn discloses that the queried handle server (“queried location server”) receives a request (“location query”) from the client for pointers associated with the handle in the request (“for a desired identifier”). (§IX.A.1.i.) As discussed for claim 3, Kahn discloses that if the queried handle server does not contain the requested pointers relevant to the desired handle (“if the queried location server

does not contain location information relevant to the desired identifier”), the handle server returns a not-responsible-for-handle message (“redirect message”) which provides information that informs the client to download an updated table from the handle server directory and recalculate the location of a correct handle server containing the handle and associated pointers (collectively, “location information”) associated with the handle in the request (“redirect message comprises information for finding a location server having location information related to the desired identifier”). (§IX.A.2; EX1002, ¶88.)

6. Claim 12

Kahn discloses claim 12. As discussed for limitation [1.pre], Kahn discloses a handle server maintains sets of handles and associated pointers (“location information”). (§IX.A.1.i.) Further, pointers include the domain name (“network address information”) of a repository storing digital objects in a network. (EX1005, Abstract, 6:8-35, 11:16-18, 21:21-24, 29:5-12, 73:23-31.) Thus, each of the multiple sets of a handle and associated pointers (collectively “location information”) includes the domain name (“network address information”) of a repository storing digital objects in a network. (EX1002, ¶89.)

7. Claim 13

Kahn discloses claim 13. As discussed for claim 12, Kahn discloses a domain name (“network address information”) of a repository (“a network address

of a data repository”) storing a digital object identified by a unique handle (“pertaining to the desired identifier”). (§IX.A.6.) As discussed for limitation [1.a], the digital object is comprised of sequences of bits/digits (“data”). (§IX.A.1.ii.) Accordingly, a domain name (“network address information”) indicates the address of a repository (“comprises a network address of a data repository”) which stores sequences of bit/digits pertaining to the object identified by a unique handle (“data pertaining to the desired identifier”) in a network. (EX1002, ¶90.)

8. Claim 14

i) [14.pre]

To the extent limiting, Kahn discloses the limitations therein. (EX1002, ¶¶91-92.) **First**, Kahn discloses a method for handling a client’s requests (“location queries”) to retrieve a digital object in a network (“network”) containing multiple handle servers (“a plurality of location servers”). (§IX.A.1.i; EX1002, ¶91.)

Second, Kahn discloses that each handle server (“location server”) stores “a unique set of location information of an aggregate set of location information” for at least the reasons discussed for limitation [10.a]. (§IX.A.5.ii.) As discussed for limitation [1.pre], Kahn discloses that the location information stored on each handle server is stored in identifier/location mappings, such that each of the

handles stored on a handle server (“each of a plurality of identifiers”) is associated with one or more pointers (“at least one location”). (§IX.A.1.i; EX1002, ¶92.)

ii) [14.a]

Kahn discloses this limitation. As discussed for limitation [1.c], Kahn discloses receiving a request (“location query”) to retrieve the location of a digital object from the client (“client”) at one of the multiple handle servers (“one of the plurality of location servers”), where the request is for an object’s location (“location query requesting an entity’s location”) in the network. (§IX.A.1.iv; EX1002, ¶93.)

Kahn discloses the claimed “client” under the AWS Proceeding’s construction. (§VIII; EX1001, 4:30-34.) For example, Kahn discloses a user entering a handle into a client (“client”) to obtain a pointer indicating the digital object’s location in the distributed network. (EX1005, 46:25-47:25, 73:1-74:32.) Kahn discloses that the user interface may be “in the form of software running on a workstation” (*id.*, 26:28-34, FIG. 2) and is part of a system that operates on a network (*id.*, 4:23-32, 19:26-31, 26:28-28:28, FIG. 2). Thus, a POSITA would have recognized that the client (“client”) is either a hardware and/or software component attached to a network. As further discussed for limitation [1.c], the client also sends a request (i.e., location request message) to a handle server (i.e., location server) to initiate lookup of mappings of handles (i.e., identifiers) to one

or more pointers (which identify a digital object's location). (§IX.A.1.iv; EX1002, ¶94.)

iii) [14.b]

Kahn discloses and/or suggests this limitation. (EX1002, ¶¶95-96.) **First**, as discussed for limitation [1.c], Kahn discloses sending a message containing a list of pointers (“sending a location response message”) to the client if the handle server that received the request (“to the client if the queried location server”) contains a handle and associated pointers (collectively “location information”) relevant to an object identified in a request (“relevant to an entity identified in the query”) to retrieve a digital object. (§IX.A.1.iv; EX1002, ¶95.)

Second, as discussed for limitation [1.c], Kahn discloses the message (“location response message”) contains a list of pointers (part of the “location information”), where the pointers in the list are associated with the handle relevant to the requested object (“entity”) identified in the request (“query”). (§IX.A.1.iv.) As discussed for limitation [1.pre], Kahn discloses and/or suggests returning the handles along with the associated pointers (collectively, “location information”) in the location response message. (§IX.A.1.i.) As discussed for limitation [10.b], each pointer identifies a repository containing a digital object, and a POSITA would have understood that Kahn's repository is an “application server” as claimed. (§IX.A.5.iii.) Accordingly, Kahn discloses and/or suggests that the

message (“location response message”) contains a list of pointers associated with the desired handle (collectively “location information”), where the pointers identify a repository (“location information identifying at least one application server”) containing the data relevant to the object identified in the request (“information relevant to the entity identified in the query”). (EX1002, ¶96.)

iv) [14.c]

Kahn discloses this limitation. (EX1002, ¶¶97-98.) As discussed for claim 3, Kahn discloses that a queried handle server sends a not-responsible-for-handle message (“redirect message”) to the client (“client”) if the queried handle server (“queried location server”) does not contain handles and associated pointers (collectively “data location information”)⁶ relevant to the object identified in the request (“relevant to the entity identified in the query”). (§IX.A.2.) As further discussed, Kahn discloses that the not-responsible-for-handle message (“redirect message”) provides information that informs the client to download an updated table from the handle server directory and recalculate the location of a correct handle server which the client may use to retrieve the object identified in the query

⁶ Kahn discloses “data location information” under the AWS Proceeding’s construction of “location information” for the same reasons as discussed for limitation [1.pre]. (§§VIII, IX.A.1.ii.)

(“redirect message comprising information for finding a location server storing the entity identified in the query”). (*Id.*; EX1002, ¶97.)

To the extent limitation [14.c] does not require that the redirect message be sent from the queried location server, Kahn also discloses this limitation in another way. Kahn discloses that the hash range/handle-server-domain-name table (“redirect message”) is downloaded by the client/requesting system (“sending...to the client”) in response to the client/requesting system receiving a not-responsible-for-handle-message. (EX1005, 47:13-19, 74:5-9.) A POSITA would have understood that when a client downloads the handle-server-domain-name table, the table would be transmitted to the client in the form of a message (“transmitting a redirect message to the first client”). For example, Kahn discloses that the “handle system is based on the UDP protocol” (*id.*, 26:8), and thus a POSITA would have recognized that information sent between servers in the handle system, e.g., between handle server directory and client, as with transmission of the handle-server-domain-name table, would have been sent as UDP messages. (EX1014, 1; EX1016, 4:37-5:12.) Further, as discussed above, this download/transmission of the handle-server-domain-name table occurs whenever the not-responsible-for-handle message is sent, i.e., “if the queried location server does not contain data location information relevant to the entity identified in the query.” The message including the handle-server-domain-name table also “compris[es] information for

finding a location server storing the entity identified in the query.” As explained for limitation [1.pre] and claim 3, the updated table associates hash ranges with handle server domain names, and allows the client to determine the location of the correct handle server that contains a pointer for retrieving the digital object associated with the handle. (§§IX.A.1.i, IX.A.2; EX1002, ¶¶98.)

9. Claim 15

Kahn discloses claim 15. (EX1002, ¶¶99.) For example, as discussed for limitation [1.c] Kahn discloses applying a handle from the client (“applying information in the location query from the client”) to a hash function (“indexing function”) to determine if the requested handle server (“queried location server”) contains the handle and associated pointers (“location information”) relevant to the object identified in the client’s request (“query”) to retrieve the location of the digital object. (§IX.A.1.iv.) As discussed for limitation [1.c], the client applies a handle to a hash algorithm, resulting in a hash code/value. (*Id.*) The hash code/value is used to index into the hash range/handle-server-domain-name table to determine the correct handle server containing the handle and associated pointers to the digital object. (*Id.*)

10. Claim 16

Kahn discloses claim 16. (EX1002, ¶¶100-101.) For example, Kahn discloses each one of the multiple handle servers (“each of the plurality of location

servers”) contains a table in a handle server mapping handles with associated pointers (“an indexed table of identifiers and associated locations”). In particular, Kahn explains “...each handle server ha[s] a table which associates handles with pointers.” (EX1005, 7:17-20, 21:8-10, 37:31-33.) A POSITA would have further understood that the table in the handle server is an “indexed table” because each handle may be used as an index to retrieve the associated one or more pointers (“locations”). (EX1002, ¶100.)

Kahn further discloses “wherein applying information in the location query comprises applying an identifier in the location query to a hash function.” As discussed for claim 15, Kahn discloses “applying information in the location query,” which involves applying a handle (“identifier”) provided in the client’s request for retrieving the location of a digital object (“location query”) to a hash function (“indexing function”) to determine if the handle server contains the handle and associated pointers. (§IX.A.9; EX1002, ¶101.)

B. Ground 2: Kahn in View of Krasner Renders Claims 2 and 11 Obvious

1. Claim 2

Kahn in view of Krasner discloses and/or suggests claim 2. (EX1002, ¶¶102-106.) Kahn discloses that handle servers store handles and associated pointers (“at least one location”) (§IX.A.1.i), but does not explicitly disclose “wherein the at least one location comprises a plurality of geographic locations,

each of the geographic locations representing a position of the entity at a different time.” However, a POSITA would have found it obvious to implement this feature in view of Krasner. (EX1002, ¶102.)

Like Kahn, Krasner relates to a client-server system. Krasner discloses a system that has location servers that store location information pertaining to a mobile GPS unit. (EX1009, Abstract, 4:10-15.) Thus, Krasner is similar to Kahn, which stores location information pertaining to digital objects. (§IX.A.1.i; EX1005, Abstract, 7:5-20, 20:31-21:2.) A POSITA would have therefore been motivated to consider Krasner when implementing Kahn’s system/method. (EX1002, ¶103.)

Krasner additionally discloses a “system for tracking the location of a mobile GPS receiver over time.” (EX1009, 3:56-57; *see also id.*, 1:48-51, 2:39-41.) The mobile GPS unit receives GPS signals at various positions and determines pseudoranges from those signals, which are stored with corresponding time stamps. (*Id.*, 5:49-53.) Krasner’s system includes a location server communicating with the mobile GPS unit. (*Id.*, 4:10-13.) The mobile GPS unit transmits the pseudoranges and time stamps to the location server when certain event(s) occur, e.g., after a certain time period or an emergency. (*Id.*, 4:66-5:24; 5:49-6:15, 6:54-7:23.) The location server may use this information to “provide

concierge services or routing information” to a user through the wireless communication system. (*Id.*, 7:23-28; EX1002, ¶104.)

A POSITA would have been motivated to include such when implementing Kahn’s system. A POSITA would have recognized that Krasner’s system would beneficially allow a user to track an object’s location over time. Thus, a POSITA would have been motivated to include features similar to Krasner’s system when implementing Kahn’s system, such that each handle server would be configured to store geographical location information pertaining to an object’s position over time. (EX1002, ¶105.)

A POSITA would have also had a reasonable expectation of success combining the teachings of Kahn and Krasner, as implementing the above-described features would have involved applications of known technologies (e.g., known systems of storing data (Kahn)) according to known methods (e.g., known methods of storing geographic location information (Krasner)) to yield the predictable result of a system that stores geographic location information on handle servers, as discussed above. (EX1002, ¶106.) *KSR*, 550 U.S. at 416.

2. Claim 11

The Kahn-Krasner combination discloses and/or suggests claim 11. (EX1002, ¶107.) As discussed for claim 2, the Kahn-Krasner combination discloses and/or suggests that a handle server stores geographical location

information (“the location information comprises geographic location”).

(§IX.B.1.)

C. Ground 3: Kahn in View of Vingralek Renders Claims 3-4, 7, 10, 12-17, 23, and 30 Obvious

1. Claim 3

Kahn in view of Vingralek discloses and/or suggests claim 3. (EX1002, ¶¶108-117.) As discussed for Ground 1, Kahn discloses claim 3. (§IX.A.2.) To the extent Kahn does not disclose “the redirect message comprising information for finding a location server known to have location information relevant to the location query,” a POSITA would have found it obvious to implement this feature in view of Vingralek. (EX1002, ¶108.)

Like Kahn, Vingralek relates to storing data in a distributed network. For example, Vingralek discloses a “distributed file organization system” where a file is “organized into buckets...spread across multiple servers” (EX1007, 4) and clients can modify the file’s distribution by issuing requests to insert/delete/retrieve records for a given key *K* (akin to an identifier of a record) in the file (*id.*, 9). Vingralek offers “a new approach for hash-based distributed files that strives for...‘cost-conscious’ scalability...to minimize the number of servers...used for providing the necessary performance capacity.” (*Id.*, 6; *see also id.*, 5-8.) Thus, Vingralek is similar to Kahn, which discloses a hash-based object distribution system and using a handle to locate/retrieve data. (§IX.A.1.i.). A POSITA would

have therefore been motivated to consider the teachings of Vingralek when implementing Kahn's system/method. (EX1002, ¶109.)

Vingralek additionally discloses propagating updates to clients' and other servers' address tables in a deferred manner. (EX1007, 6.) To limit communication overhead, clients learn about changes to the hash/address table only when a client request reaches a wrong server, in which case the server forwards the client's request to the correct server and provides updates to the client's address table. (*Id.*, 5-6; EX1002, ¶110.)

Vingralek's method of deferred address table updates operates as follows. A hash function is applied to a key, which is used as an index to an address table listing available servers. (EX1007, 9.) Each client and server has its own copy of the address table, but each table may have stale information if changes have been made to the servers since the table was last updated. (*Id.*, 10.) When a client seeks to retrieve records associated with a given key, it applies the hash function to the key, identifies the server where that key is located by using the hashed key as an index into the current copy of the address table, and then sends the key to that first server. (*Id.*, 9-10.) The first server that receives the key also applies the hash function to the key and identifies in its copy of the address table the server where the key resides. (*Id.*, 10-11.) If the key was indeed sent to the correct server, then that server performs the requested operation. (*Id.*, 11.) But if the key was sent to

the wrong server, the first server forwards the request to another server it has identified through applying the hash function, and sends a message to the client to update the client's address table. (*Id.*, 11-13, 25.) Thus, Vingralek discloses that if a first server does not have the requested information, a message ("redirect message") is sent to the client with information to update the client's address/hash table, which the client uses when calculating the location of a server. (EX1002, ¶¶111-113.)

A POSITA would have been motivated to include features similar to those disclosed by Vingralek when implementing Kahn by providing a deferred hash table update to the client (for the client to calculate a location of a correct handle server). Like Vingralek, Kahn discloses that both client and handle server store copies of the hash table. (EX1005, 38:9-17, 46:29-32.) Further, Kahn discloses its data/file distribution is subject to change, e.g., when the data and/or the data repository have changed. (*Id.*, 20:29-33, 33:3-14, 45:10-12.) Accordingly, Kahn's hash table is also subject to change. (*Id.*, 46:22-34, 74:5-9.) Thus, consistent with Vingralek, Kahn's system would have benefited from implementing such a hash table update, as it would have allowed the client to use a correct/updated hash table to locate a handle server for data retrieval. (EX1002, ¶¶114-115.) Further, Vingralek explains that this approach would have decoupled the communication overhead from the number of servers/clients in the system and improved

scalability, consistent with Kahn's goal and expectation of scaling. (EX1005, 9:23-27, 33:3-14, 46:22-34; EX1007, 4; EX1002, ¶¶114-115.)

Additionally, a POSITA would have been motivated to implement the deferred hash table updates in a system similar to Kahn, e.g., by modifying Kahn's mechanism of notifying the client. As disclosed in Kahn, the client would apply the hash function to its downloaded copy of the hash table, and send a request to the first handle server. While Kahn discloses that "[t]he handle server verifies that the handle falls within the set of handles for whose hash values it is responsible," it does not explicitly disclose the specifics of its verification method. (EX1005, 73:32-34.) Thus, a POSITA would have been motivated to use a verification method similar to that of Vingralek. (*Id.*, 38:9-17, 46:29-31.) Thus, in the Kahn-Vingralek combination, for example, the first handle server ("queried location server") would apply the hash function to the handle and use the result as an index into the hash table as a means of determining whether it has the requested location information. If the first handle server does not have the requested location information, i.e., the handles and associated pointers for the requested object ("if the queried location server lacks location information for the desired entity"), the handle server would transmit a message ("redirect message") to the client, including the handle server's hash table for use by the client to calculate a location of a correct handle server that contains the pointers associated with the handle

(“redirect message comprising information for finding a location server known to have location information relevant to the location query”). The client would then perform the calculation of the location of the correct handle server (“finding a location server known to have location information relevant to the location query”) and send the request to that correct handle server. (§IX.A.2.) A POSITA would have understood that the above-described implementation, where the client calculates a correct handle server and forwards the request to that server (e.g., as in Kahn), would have been one of only several options, i.e., calculating and forwarding at either the client or the handle server that processes the information request, and thus would have been one of a “finite number of identified, predictable solutions.” *Perfect Web Techs., Inc. v. InfoUSA, Inc.*, 587 F.3d 1324, 1331 (Fed. Cir. 2009). A POSITA would have further understood that implementing such a redirect message would allow hash table updates to propagate from handle server to client, while reducing the number of downloads performed from the handle directory servers (i.e., by reducing the number of downloads by the client). (EX1002, ¶116.)

A POSITA would have also had a reasonable expectation of success combining the teachings of Kahn and Vingralek. (EX1002, ¶117.) Both references are in the same field of art and both relate to hash-based distributed computing/file systems. (*See supra.*) Moreover, as discussed above, Kahn already

utilizes a message to provide updated information to the client when a wrong server is reached, but would benefit from reducing communication overhead by implementing Vingralek's teaching. Such a modification would have involved applications of known technologies (e.g., known distributed data storage systems (Kahn)) according to known methods (e.g., known methods of deferred updates for hash tables using a redirect message (Vingralek)) to yield the predictable result of a distributed data storage system that limits communication overhead and improves scalability, as discussed above. *KSR*, 550 U.S. at 416.

2. Claim 4

The Kahn-Vingralek combination discloses and/or suggests claim 4. (EX1002, ¶¶118-120.) For the same reasons discussed for claim 3, the Kahn-Vingralek combination discloses and/or suggests “wherein the information for finding the location server known to have location information relevant to the location query.” (§IX.C.1.) For example, the Kahn-Vingralek combination discloses and/or suggests a hash range/handle-server-domain-name table containing information for finding the correct handle server. (*Id.*; EX1002, ¶118.)

Additionally, the Kahn-Vingralek combination discloses and/or suggests “wherein the information...comprises a list of location servers and a corresponding list of each of the entities having location information on the location servers.” Kahn discloses that the hash range/handle-server-domain-name table comprises a

list of handle servers (“location servers”), as shown in Figure 7 below. (EX1005, FIG. 7, 38:9-17, 46:22-24, 47:1-21, 73:17-27.)

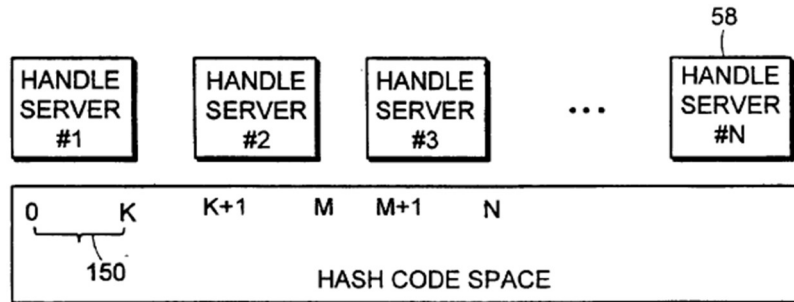


FIG. 7

(*Id.*, FIG. 7.) Consistent with as shown in Figure 7, a POSITA would have understood that the updated hash range/handle-server-domain-name table likewise includes a “list” of handle servers (“a list of location servers”). (EX1005, FIG. 7; EX1002, ¶119.)

Furthermore, Kahn explains that each handle server domain name in the hash range/handle-server-domain-name table corresponds to a hash range. (EX1005, 46:22-24, 47:1-5, 73:17-20.) Kahn explains that a hash range is a list of individual hash codes/values, i.e., the output of a hash algorithm applied to a handle. (*Id.*, 46:22-24, 47:1-5, 73:17-20, FIG. 7.) Since each handle is associated with an object (“entity”), a POSITA would have understood that the list of handle servers (“a list of location servers”) corresponds to a list of hash codes/values identifying each of the objects (“each of the entities”) having an identifying handle

and associated one or more pointers (collectively, “location information”) on the handle servers (“location servers”). (EX1002, ¶120.)

3. Claim 7

The Kahn-Vingralek combination discloses and/or suggests claim 7. (EX1002, ¶¶121-122.) **First**, as discussed for limitation [1.c], Kahn discloses a client’s request (“location query”) to retrieve the location of a digital object. (§IX.A.1.iv.) This request, as discussed for limitation [1.pre], includes a client sending a handle identifying the requested object (“location query identifying the desired entity”) to a handle server. (§IX.A.1.i; EX1005, 46:34-47:8, 73:8-22.) As further discussed for limitation [1.pre], each handle uniquely identifies an object (“unique identifier for the desired entity”). (§IX.A.1.i; EX1005, 1:16-20, 21:3-5; EX1002, ¶121.)

Second, as discussed for limitation [1.c], Kahn discloses programming logic on a handle server (“location server”) that is configured to perform a handle-to-pointer translation service. (§IX.A.1.iv.) As discussed for Ground 1, claim 6, and Ground 3, claim 3, in the Kahn-Vingralek combination, a queried handle server (“location server”) is configured to apply the hash function (“index function”) to the handle in the request (“unique identifier”) as part of the process for retrieving the handle and associated pointers (collectively, “at least a portion of the location information”) associated with the handle (“unique identifier”) in the request, where

the handles and pointers are stored in a table (“indexed location store”) on the handle server. (§§IX.A.4, IX.C.1.) In particular, the handle server applies the hash function to determine if it has the requested pointers associated with the request. (§IX.C.1.) A POSITA would have recognized that an index function is a function that returns an index to a table, and thus a hash algorithm, which returns a value that is used as an index to the hash table, is an “index function.” (EX1002, ¶122.)

4. Claim 10

As discussed for Ground 1, Kahn discloses and/or suggests claim 10. (§IX.A.5.i-iii.) To the extent that Kahn’s redirect message described for limitation [10.b] does not disclose that the “redirect message comprises information for finding a location server having location information related to the desired identifier,” a POSITA would have been motivated and found it obvious to implement this feature in view of Vingralek for reasons similar to those as discussed for claim 3 in Ground 3. (§IX.C.1; EX1002, ¶123.) For example, as discussed for claim 3 in Ground 3, the Kahn-Vingralek combination discloses and/or suggests that after the first handle server (“queried location server”) receives the request (“location query”) to retrieve pointers associated with a handle provided in the request, the handle server applies the hash function to the handle and to its own copy of the hash table to determine whether it has the requested pointers associated with the handle. (§IX.C.1.) As further discussed, if the first

handle server does not have the requested handle and associated pointers relevant to the handle in the request (“if the queried location server does not contain location information relevant to the desired identifier”), the handle server would send a message to the client (“responsive to a location query for a desired identifier to return...a redirect message”) to update the client’s hash table, thus allowing the client to use the updated hash table to calculate the correct handle server that it may send the request to (“wherein the redirect message comprises information for finding a location server having location information related to the desired identifier”). (*Id.*; EX1002, ¶123.)

5. Claim 12

Kahn discloses claim 12 for the same reasons as discussed for Ground 1, claim 12. (§IX.A.6; EX1002, ¶124.)

6. Claim 13

Kahn discloses claim 13 for the same reasons as discussed for Ground 1, claim 13. (§IX.A.7; EX1002, ¶125.)

7. Claim 14

As discussed for Ground 1, Kahn discloses and/or suggests claim 14. (§§IX.A.8.i-iv.) To the extent that Kahn’s redirect message described for limitation [14.c] does not disclose the “redirect message comprising information for finding a location server storing the entity identified in the query,” a POSITA

would have been motivated and found it obvious to implement this feature in view of Vingralek for reasons similar to those as discussed for claim 3 in Ground 3.

(§IX.C.1; EX1002, ¶126.) For example, as discussed for claim 3 in Ground 3, the Kahn-Vingralek combination discloses and/or suggests that after the first handle server (“queried location server”) receives the request (“query”) to retrieve pointers associated with a handle provided in the request, the handle server applies the hash function to the handle and to its own copy of the hash table to determine whether it has the requested pointers associated with the handle. (§IX.C.1.) As further discussed, if the first handle server does not have the requested handle and associated pointers relevant to the object identified in the request (“if the queried location server does not contain data location information relevant to the entity identified in the query”), the handle server would send a message to the client (“sending a redirect message to the client”) to update the client’s hash table, thus allowing the client to use the updated hash table to calculate the correct handle server that it may send the request to (“the redirect message comprising information for finding a location server storing the entity identified in the query”). (*Id.*)

8. Claim 15

To the extent claim 15 is found to require that the location server is the component that performs the recited application of an indexing function, a

POSITA would have been motivated and found it obvious to implement this feature in view of Vingralek for reasons similar to those discussed above.

(§IX.C.1; EX1002, ¶127.) Indeed, as discussed for claim 3, in the Kahn-Vingralek combination, a queried handle server (“location server”) is configured to apply the handle identified in the request (“location query”) from the client (“applying information in the location query from the client”) to a hash function (“indexing function”) to determine whether the handle server contains the handle and associated pointers (collectively, “location information”) associated with the handle (identifying an object (“entity”)) that is included in the request (“to determine if the queried location server contains location information relevant to the entity identified in the query”). (§IX.C.1) As discussed for claim 7, Kahn’s hash function is an “indexing function.” (§IX.C.3.)

9. Claim 16

As discussed for Ground 1, Kahn discloses claim 16. (§IX.A.10; EX1002, ¶128.) To the extent claim 16 is found to require that the location server is the component that performs the recited application of an indexing function (“wherein applying information in the location query comprises applying an identifier in the location query to a hash function”), a POSITA would have been motivated and found it obvious to implement this feature in view of Vingralek for reasons similar to those discussed for Ground 3, claim 3. (§IX.C.1.) As discussed for Ground 3,

claim 15, the Kahn-Vingralek combination discloses and/or suggests that “applying information in the location query” comprises applying a handle (“identifier”) in the request (“location query”) to a hash function. (§IX.C.8.)

10. Claim 17

i) [17.pre]

To the extent limiting, Kahn in view of Vingralek discloses and/or suggests the limitations therein. (EX1002, ¶¶129-135.) As discussed for limitation [1.pre], Kahn discloses a system with a plurality of handle servers (“a system having a plurality of location servers”) for storing and retrieving handles and their associated pointers (collectively, “location information”). (§IX.A.1.i.) Kahn also discloses that each handle server (“each of the plurality of location servers”) stores “a unique set of location information of an aggregate set of location information” for the reasons discussed for Ground 1, limitation [10.a]. (§IX.A.5.ii.)

Kahn does not explicitly disclose “a method of scaling at least one of capacity and transaction rate capability in a location server” in the system. A POSITA would have been motivated and found it obvious to implement this feature in view of Vingralek. (EX1002, ¶130.)

Vingralek and Kahn are in the same field of art, and a POSITA would have been motivated to consider the teachings of Vingralek when implementing Kahn’s system/method. (§IX.C.1; EX1002, ¶131.) Vingralek discloses a system where a

file is organized into buckets spread across servers. (§IX.C.1; EX1007, 4.)

Vingralek employs a local load control at each server, preventing server overloading by redistributing buckets when a specified server utilization is reached. (EX1007, 6-17, 12.)

Vingralek describes two load capacity concepts: the “feasible load capacity,” i.e., “the maximum number of keys that the server can keep without being overloaded,” and the “panic load capacity,” i.e., “the number of keys after which a server is no longer capable of accepting additional keys.” (*Id.*, 10.) Vingralek describes several load management techniques for splitting/migrating buckets, and for moving buckets to an existing server or acquiring a new server. (*Id.*, 11-14.) For example, if a server exceeds its feasible load capacity, the file advisor begins monitoring that server’s capacity (*id.*, 12) until the server reaches the panic load capacity, upon which the file advisor identifies a second server having enough unused performance capacity to accept some buckets from the first at-capacity server, and sends a migrate message to the first server, triggering the transfer of buckets from the first server to another (second) server. (*Id.*, 13; EX1002, ¶¶132-133.)

A POSITA would have been motivated to include scalability features similar to those disclosed by Vingralek when implementing a system similar to that of Kahn to manage the handle servers’ load capacities while minimizing the number

of servers storing location data, so as to reducing the cost of scaling up the distributed system. (EX1002, ¶¶133-134.) For example, when a handle server is no longer able to accept additional requests to add handles (e.g., reaching panic load capacity as discussed in Vingralek), a POSITA would have found it obvious to move/transfer certain handles and associated pointers (“associated locations”), to a second/another handle server to allow the system of handle servers to continue fulfilling requests. Additionally, a POSITA would have found it beneficial to implement Vingralek’s teaching as it strives for the objective of cost-conscious scalability by not acquiring new servers unless the load on existing servers exceeds certain thresholds. (EX1007, 10-13.) Moreover, a POSITA would have found Vingralek’s method beneficial because while Kahn discloses that the digital objects, and thus handle servers storing handles identifying those objects, in a network can be scaled arbitrarily, it does not explicitly disclose the specific processes for scaling that system of servers, e.g., how or when a server’s load capacity is exceeded, which Vingralek teaches. (EX1005, 9:23-27; 33:8-10.)

A POSITA would have had a reasonable expectation of success combining the teachings of Kahn and Vingralek. (EX1002, ¶135.) As discussed above, both references are in the same field of art. (*See supra.*) Moreover, Kahn itself suggests that the digital objects, and thus handle servers, may be scaled arbitrarily. (EX1005, 9:23-27, 33:8-10.) Thus, combining the teachings of Kahn and

Vingralek would have involved modifying the Kahn system in a manner that Kahn suggests is feasible. Such a combination would have involved applications of known technologies (e.g., known distributed data storage systems (Kahn)) according to known methods (e.g., known methods of scaling distributed systems (Vingralek)) to yield the predictable result of a system that implements a method of scaling the handle servers in a network, as discussed above. *KSR*, 550 U.S. at 416.

ii) [17.a]

Kahn discloses this limitation. (EX1002, ¶136.) For example, Kahn discloses “location information specifying the location of information related to the identifier.” As discussed for limitation [1.pre], one or more handles and their associated pointers (collectively, “location information”) specify the location of digital objects related to the handle (“the location of information related to the identifier”). (§IX.A.1.i.) Further, Kahn discloses “providing a transfer protocol configured to transport identifier and location information.” For example, Kahn discloses the handle system is “based on the UDP protocol,” which “enables a large number of transactions to be handled efficiently.” (EX1005, 26:8-10, FIG. 33.) Thus, a POSITA would have understood that the UDP protocol (“transfer protocol”) is used to transport one or more handles and their associated pointers (where the “identifier” is a part of the collective “location information”).

iii) [17.b]

Kahn discloses this limitation. (EX1002, ¶¶137-138.) Consistent with as discussed for [1.a], Kahn discloses storing one or more handles and associated pointers (collectively, “location information”) at a first handle server (“first location server”). (§IX.A.1.ii)

Further, Kahn discloses “storing location information **formatted according to the transfer protocol** at a first location server.” As discussed for limitation [17.a], Kahn discloses a handle system based on a UDP protocol. (§IX.C.10.ii.) A POSITA would have understood that because Kahn’s handle system is based on a UDP protocol, the handle servers in that system (including “first location server”) would also store one or more handles and their associated pointers (collectively, “location information”) in accordance with UDP protocol (“transfer protocol”) for each of communicating within the system using the same protocol. (EX1002, ¶138.)

iv) [17.c]

Kahn discloses this limitation. (EX1002, ¶¶139-141.) For example, Kahn discloses receiving a handle (“identifier”) and an associated pointer (“location relevant to the identifier”) at a first handle server (“first location server”) when inserting a new digital object into the system. Kahn discloses that when a digital object is deposited in a repository, it is made available in the handle system.

(EX1005, 20:4-14, 37:4-6.) In particular, a handle for the digital object to be deposited and a pointer associated with the handle are both received at a handle server. (*Id.*, 20:15-17, 39:34-40:1, 43:31-35, 45:10-21.) Thus, Kahn discloses receiving a handle (“identifier”) and a pointer related to the handle (“location relevant to the identifier”) at a first handle server (“a first location server”).

(EX1002, ¶139.)

To the extent Kahn does not explicitly disclose a handle server receiving both a handle and associated pointers for insertion, a POSITA would have found it obvious to include such a feature in Kahn. (EX1002, ¶140.) A POSITA would have found it beneficial to implement this feature because sending the handle (“identifier”) and associated pointers (“location relevant to the identifier”) in a single request would allow for combining handle insertion and storage processes. Indeed, Kahn discloses processes for inserting handles and associated information (e.g., pointers) into a handle server (EX1005, 20:4-17, 37:4-6, 39:34-40:1, 43:31-35, 45:10-21), and further discloses that the information is stored on the handle server in a table that associates handles with pointers (*id.*, 7:17-20, 37:31-33). A POSITA would have understood that combining the insertion and storage processes, such that already-associated handles and pointers are sent in a single request and received at the handle server, would have reduced processing steps and

communication overhead since additional steps (beyond an insertion request) would not be required to associate handles and pointers on the handle server.

A POSITA would have also had a reasonable expectation of success in implementing this modification. Kahn discloses that handles and associated information may be sent to a handle server (EX1005, 20:4-17, 37:4-6, 39:34-40:1, 43:31-35, 45:10-21), and thus the handle server would be readily able to receive an insertion request containing a handle and associated pointers. Moreover, Kahn is already capable of storing handles along with associated pointers (*id.*, 7:17-20, 37:31-33). Such a modification would have involved applications of known technologies (e.g., known handle server storing a table that associates handles with pointers) according to known methods (e.g., known handle server receiving a request to insert handles and associated information) to yield the predictable result of a handle server receiving a handle and associated pointers for insertion, as discussed above. (EX1002, ¶141.) *KSR*, 550 U.S. at 416.

v) [17.d]

Kahn discloses this limitation. (EX1002, ¶142.) As discussed for claim 6, Kahn discloses that each handle server (“location server”) has a table in the handle server associating handles with pointers (“location store”) where the handles in the table (“the location store comprising a plurality of identifiers”) are each associated with one or more pointers (“each identifier associated with at least one location”).

(§IX.A.4.) A POSITA would have thus understood that when a handle server (“first data location server”)⁷ stores the received handle (“the received identifier”) and associated pointer (“the received location”), the handle server would store the information in the table (“location store”) such that the handle is associated with the corresponding pointer (“storing the received location in a location store at the first data location server...wherein the received location is associated with the received identifier in the location store”).

vi) [17.e]

The Kahn-Vingralek combination discloses and/or suggests this limitation. (EX1002, ¶143.) As discussed for limitation [17.pre], in the Kahn-Vingralek combination, if a handle server (“first data location server”) reaches the panic load capacity (“when a performance criterion of the first location server reaches a predetermined performance limit”), it would have been obvious to move/transfer certain handles and associated pointers to a second/another handle server to allow the system of handle servers to continue fulfilling requests. (§IX.C.10.i.) Thus, the Kahn-Vingralek combination discloses and/or suggests transferring a portion of

⁷ Kahn discloses a “data location server” under the AWS Proceeding’s construction of a “location server” for the same reasons as discussed for limitation [1.pre].

(§§VIII, IX.A.1.ii.)

the handles (“identifiers”) and associated pointers (“locations”) to the second handle server (“transferring a portion of the identifiers and associated locations to a second data location server”) when the panic load capacity is reached (“when a performance criterion of the first location server reaches a predetermined performance limit”).

11. Claim 23

The Kahn-Vingralek combination discloses and/or suggests this limitation. (EX1002, ¶144.) As discussed for limitations [17.pre] and [17.e], the panic load capacity (“performance criterion”) is “the number of keys after which a server is no longer capable of accepting additional keys.” (EX1007, 10, 21; §§IX.C.10.i, IX.C.10.vi.) While Vingralek expresses this capacity as a number of keys (similar to Kahn’s handles, i.e., identifiers) that a server may hold, Vingralek explains that it assumes that the key size is a fixed/constant amount of storage space. (EX1007, 21, 22.) Thus, a POSITA would have recognized that in the Kahn-Vingralek combination the panic load capacity factors in and/or represents the amount of persistent storage space available on a handle server (“the performance criterion comprises an amount of available persistent storage space in the first location server”).

12. Claim 30

The Kahn-Vingralek combination discloses and/or suggests this limitation. (EX1002, ¶145.) As discussed for limitation [17.pre], Vingralek discloses that after a server reaches the feasible load capacity, the file advisor begins monitoring that server to determine if it hits other thresholds, e.g., the panic load capacity (“monitoring the performance criterion”). (§IX.C.10.i.) When a server hits the panic load capacity (“when the first location server reaches the predetermined limit”), the file advisor begins identifying a second server for transferring buckets/information to, and sending a migrate message to the first server to trigger transferring of the information (“automatically transferring” information “when the first location server reaches the predetermined limit”). (*Id.*) For reasons similar to those discussed for limitation [17.pre], a POSITA would have found it obvious to implement in the Kahn-Vingralek combination processes similar to those described above, to monitor the handle server’s load capacity and automatically transfer the handles (“the portion of identifiers”) and associated pointers (“associated locations”)) when the monitored server reaches the panic load capacity (“when the first location server reaches the predetermined limit”).

D. Ground 4: Kahn in View of Vingralek and Krasner Renders Claim 11 Obvious

1. Claim 11

The Kahn-Vingralek combination in view of Krasner discloses and/or suggests claim 11. (EX1002, ¶146.) As discussed for Ground 3, claims 3 and 10, a POSITA would have found it obvious to combine teachings of Kahn and Vingralek. (§§IX.C.1, IX.C.4.) For similar reasons to those discussed for claim 2, a POSITA would have found it obvious to combine the Kahn-Vingralek combination with Krasner. (§IX.B.1.) As further discussed for claim 2, Krasner discloses that a location server stores geographical location information (“the location information comprises geographic location”). (*Id.*) Accordingly, the Kahn-Vingralek-Krasner combination discloses and/or suggests claim 11.

E. Ground 5: Kahn in View of Vingralek and Sato Renders Claims 24-26 Obvious

1. Claim 24

Kahn in view of Vingralek and Sato discloses and/or suggests claim 24. (EX1002, ¶¶147-151.) In particular, although the Kahn-Vingralek combination does not disclose scaling a location server’s capability where the performance criterion comprises a transaction rate limit, a POSITA would have found it obvious to implement this feature in view of Sato. (EX1002, ¶147.)

Like Vingralek, Sato describes a “client-server system built on a distributed system” where the client issues requests to server(s). (EX1011, ¶[0003].) Sato’s system uses another set of servers, i.e., service location servers, which provide location information of the specific server that provide services to the client. (*Id.*,

¶[0003].) Sato discloses a server partitioning method that partitions the service range and manages location information in a distributed manner “when the load on the service location server exceeds a predetermined threshold.” (*Id.*, ¶[0005].) A POSITA would have therefore been motivated to consider Sato when implementing the Kahn-Vingralek combination. (EX1002, ¶148.)

Sato additionally discloses that each “service location server” includes a “load measuring means” measuring a server’s “service load information” (EX1011, ¶[0009]), such as “the maximum number of service executions per unit time” (*id.*, ¶[0012]). Whether to partition the server “is determined by comparing the maximum number of service executions [per unit time]...with [a predetermined] threshold value” (*id.*, ¶¶[0013], [0029]), and if partitioning is necessary, then information on the server is migrated to another server (*id.*, ¶¶[0029]-[0030]). Thus, Sato discloses a method of scaling transaction rate capability in a server, where information is transferred from a first to second server when the maximum number of service executions per unit time (or execution rate) reaches a predetermined threshold value (“performance criterion comprises a transaction rate limit”). (EX1002, ¶149.)

A POSITA would have been motivated to include features similar to those disclosed by Sato when implementing the Kahn-Vingralek combination. (EX1002, ¶150.) For example, while Vingralek generally describes a homogenous system

(EX1007, 21), its approach is “applicable also to heterogeneous servers where servers...hav[ing] different load capacities” (*id.*, 26). Thus, a POSITA would have been motivated to modify the Kahn-Vingralek system to account for heterogeneous servers where the server instruction rate (processing speed) varies based on the characteristics of each server. Sato’s method of partitioning a server based on a maximum service execution rate is advantageous because it allows for dynamic distributions of server workload upon reaching a predetermined threshold, rather than requiring distribution in advance, thereby “increasing the flexibility of the system.” (EX1011, ¶[0073].) Sato’s method would have been beneficial for allowing implementation of a partitioning method in a system of heterogeneous servers, where processing speed may vary. In such a combined system, each handle server would be configured to have a set maximum service execution rate threshold such that, in addition to transferring identifier/location mappings to a second handle server based on the load capacity principles discussed in Vingralek (e.g., panic load capacity), it would also transfer identifier/location mappings to a second handle server when a first server’s maximum service execution rate exceeds a predetermined threshold. (EX1002, ¶150.)

A POSITA would have also had a reasonable expectation of success combining the teachings of Kahn, Vingralek, and Sato. (*Id.*, ¶151.) As discussed above, both references are in the same field of art. Moreover, it would have

involved modifying the Kahn-Vingralek system in a manner that Vingralek suggests is feasible—implementing methods of partitioning servers in a heterogeneous system. Such a combination would have involved applications of known technologies (e.g., known system for partitioning servers (Vingralek)) according to known methods (e.g., known methods of partitioning when a maximum service execution rate exceeds a threshold (Sato)) to yield the predictable result of a system in which handle servers are configured to transfer information to other servers when a transaction rate limit is reached, as discussed above. *KSR*, 550 U.S. at 416.

2. Claim 25

The Kahn-Vingralek-Sato combination discloses and/or suggests claim 25. (EX1002, ¶152.) As discussed for limitation [1.c], each handle server may be a computer, and thus a POSITA would have understood that a handle server includes a “processor.” (§IX.A.1.iv.) As discussed for claim 24, the performance criterion is a maximum service *execution* rate (“transaction rate limit”). (§IX.E.1.) Thus, a POSITA would have understood that the maximum service execution rate reflects the maximum speed at which a processor of a handle server operates (“transaction rate limit comprises a processor speed limit”).

3. Claim 26

The Kahn-Vingralek-Sato combination discloses and/or suggests claim 26. (EX1002, ¶¶153-155.) Vingralek discloses the **performance** of the distributed network is “characterized by the network latency *NL* (i.e., time required to process each packet, independently of the packet size) and the **bandwidth** *BW* (i.e., transmission time needed to send one byte through the network).” (EX1007, 22.) A POSITA would have understood that the network bandwidth is of a finite value/number. Thus, Vingralek discloses “a network connection bandwidth limit.” A POSITA would have understood that the disclosed “network connection bandwidth limit” is a factor of the “transaction rate capability” (recited in limitation [17.pre]) or “transaction rate limit” (recited in claim 26) of a handle server (“location server”) because a network’s bandwidth impacts the rate at which the servers may receive requests from the client, and thus server(s)’ maximum service execution rate. (§IX.C.10.i; EX1002, ¶153.)

With the above, a POSITA, when implementing the Kahn-Vingralek-Sato combination would have found it obvious and been motivated to “transfer[] a portion of the identifiers and associated locations to a second data location server when a performance criterion of the first location server reaches a predetermined performance limit” (limitation [17.e]), where the “performance criterion” is “the transaction rate limit compris[ing] a network connection bandwidth limit” (claim

26). (§IX.C.10.vi; EX1002, ¶154.) The bandwidth of the part of the network connected to a first handle server may be limited/reduced due to, e.g., available bandwidth being occupied by an increased number of client requests. This is particularly true as the servers may be heterogeneous having different load capacities as disclosed in Vingralek. (EX1007, 26; §IX.E.1.) Thus, a POSITA would have found it beneficial to conduct a process similar to as described for limitation [17.e], e.g., identifying another handle server that has sufficient network connection bandwidth to accept buckets from a first handle server and transferring the same. (§IX.C.10.vi.) It would have reduced the load of the first handle server, and increased the second handle server’s utilization, consistent with Vingralek’s “objective of ‘cost-conscious’ scalability, by aiming to minimize the number of servers...used for providing the necessary performance capacity.” (EX1007, 6; EX1002, ¶154.)

A POSITA would have also had a reasonable expectation of success in implementing such features in the Kahn-Vingralek-Sato combination. (EX1002, ¶155.) It would have involved modifying the combination in a manner Vingralek suggests is feasible—implementing methods of partitioning servers in a heterogeneous system. The modification would have involved applications of known technologies (e.g., known system for partitioning servers and known network bandwidth limit (Vingralek)) according to known methods (e.g., known

methods of partitioning when a maximum service execution rate exceeds a threshold (Sato)) to yield the predictable result of a communication system in which handle servers are configured to transfer information to other servers when a network connection bandwidth limit is reached. *KSR*, 550 U.S. at 416.

X. DISCRETIONARY DENIAL IS NOT APPROPRIATE

A. The Board Should Not Deny Institution Under § 325

Discretionary denial under § 325(d) is not appropriate here because this petition does not present the same or substantially the same prior art or arguments previously presented to the Office during prosecution. (EX1004; EX1001, Cover.) Moreover, there are material differences between the art/arguments here and those during prosecution (*e.g.*, EX1004, 305-314), including features believed to be missing (*id.*, 334-342). Discretionary denial is also not warranted in view of the completed *ex parte* reexamination proceedings and IPR, since this petition does not present the same or substantially the same prior art or arguments presented in those proceedings, and there are material differences between the art/arguments here with those presented in those proceedings (*e.g.*, EX1042, 23-24; EX1043, 12-14; EX1047, 15-18), including features believed to be missing (EX1022, 19-23; EX1045, 9-10; EX1049, 8-10.)

Discretionary denial is also not warranted in view of the pending *ex parte* reexamination initiated by a different party, AWS. (EX1051; §II.) First, Petitioner

had no involvement with the reexamination, and Petitioner developed the present petition independently of AWS.

Second, this petition and the reexamination are based on materially different grounds and positions. (EX1051, 17-43; §§IV, IX.)

Third, the pending claims of the reexamination and the challenged claims in this IPR are not coextensive. (§§II, IV; EX1051.)

Fourth, it is PO that decided to accuse separate companies of infringement, each with separate interests in challenging the patent. And, given the statutory deadline to file an IPR, Google could not have waited until the conclusion of the reexamination. *Mueller Sys., LLC v. Rein Tech, Inc.*, IPR2020-00100, Paper 9 at 12-14 (P.T.A.B. May 12, 2020).

Fifth, the reexamination remains at an early stage, which counsels against discretionary denial. *Mueller* at 13.

B. Discretionary Denial Under *General Plastic* Is Not Appropriate

The Board has set forth seven factors in determining whether to exercise discretion to deny a petition due to an earlier proceeding. *Gen. Plastic Indus. Co., Ltd. v. Canon Kabushiki Kaisha*, IPR2016-01357, Paper 19 at 15-19 (Sept. 6, 2017) (precedential) (“*General Plastic*”).

With respect to the first factor, Petitioner is not a party, real party-in-interest, or privy to the other Patent Office proceedings and this is Petitioner’s first

challenge to the '978 patent, which “weighs especially heavily against a discretionary denial.” *Unified Patents, Inc. v. Certified Measurement, LLC*, IPR2018-00548, Paper 7 at 7-8 (PTAB Sept. 5, 2018). Moreover, AWS and Petitioner were sued at different times—four and a half years apart—and the AWS-challenged claims (§II) and the challenged claims in this IPR are not coextensive. *Valve Corp. v. Elec. Scripting Prods., Inc.*, IPR2019-00062, Paper 11 at 10 (PTAB April 2, 2019) (precedential).

The second to fifth factors “bear little relevance” here because Petitioner has never before challenged the '978 patent. *Unified* at 7-8. Petitioner has no relationship to the earlier challenges to the '978 patent. And there is no indication that AWS was aware of the Kahn-based prior art combinations at issue here.

As to the sixth factor, while the Board has finite resources, instituting this petition would be no more a burden on these finite resources than instituting any other petition. And for the seventh factor, there is no readily identifiable roadblock for the Board to issue a final determination within the statutory one-year limit.

C. The Board Should Not Deny Institution Under *Fintiv*

The merits of Petitioner’s arguments are compelling and the evidence in support is substantial. That “alone demonstrates that the PTAB should not discretionarily deny institution under *Fintiv*.” (EX1036, 4-5.) In any event, the six *Fintiv* factors do not justify denying institution.

The **first factor** (stay) weighs strongly against denial. Although Google has not yet moved to stay in district court pending IPR, the Northern District of Illinois has stayed a number of cases pending IPR proceedings since 2020. (EX1033, 1, 2, 5, 6, 10.)

The **second factor** (trial proximity) weighs strongly against denial, since “the median time-to-trial is...after the projected...final written decision.” (EX1036, 9.) Specifically, the median time-to-trial in the Northern District of Illinois is close to 57 months. (EX1035, 47.) The district court case was filed on June 30, 2023, so trial based on the statistics would occur in April 2028. On the other hand, with this petition filed in June 2024, a final written decision may be expected around December 2025. Moreover, this petition was diligently filed shortly after PO served its infringement contentions identifying asserted claims on April 25, 2024. (EX1034, 1.)

The **third factor** (investment in parallel proceedings) also weighs strongly against denial. The district court case is in the early stages. (*Id.*, 1-2.) Moreover, the district court’s discovery process is currently stayed pending resolution of Google’s Motion to Dismiss until July 15, 2024. (EX1052, 1; EX1055, 1.)

The **fourth factor** (overlap) also weighs against denial, since Petitioner challenges more claims than PO may ultimately assert in its final contentions in the

district court litigation, i.e., 10 claims per patent and 20 claims overall from the '640, '170, and '978 patents. (EX1053, 10.)

The **sixth** factor (other circumstances) also weighs against denial as the merits of Petitioner's arguments are compelling. While the **fifth** factor (parties) may weigh slightly in favor of denial because the parties are the same as in district court, that is true in almost every IPR, and under the "holistic view" of whether integrity of the system and efficiency is best served, institution is favored.

Samsung Elecs. Co. Ltd. v. Dynamics Inc., IPR2020-00505, Paper 11 at 15 (Aug. 12, 2020).

XI. CONCLUSION

Petitioner requests institution of IPR for the challenged claims based on the grounds specified in this petition.

Respectfully submitted,

Dated: June 28, 2024

By: /Naveen Modi/
Naveen Modi (Reg. No. 46,224)
Counsel for Petitioner

XII. APPENDIX A (CLAIM LISTING)

Claim 1

[1.pre] “A system having a plurality of location servers for managing location information and providing location information to location queries, the system comprising:”

[1.a] “a first location server containing a first set of location information corresponding to at least one entity, the location information comprising an identifier and at least one location string associated with the identifier, wherein the identifier identifies an entity and the location string specifies a location of data pertaining to the entity”

[1.b] “a second location server comprising a second set of location information, wherein at least a portion of the second set of location information differs from the first set of location information; and”

[1.c] “programming logic stored on each of the location servers responsive to a location query identifying a desired entity to return a location message, the location message comprising at least one location of data pertaining to the desired entity, if the location server

receiving the location query contains location information for the desired entity.”

Claim 2

“The system of claim 1, wherein the at least one location comprises a plurality of geographic locations, each of the geographic locations representing a position of the entity at a different time.”

Claim 3

“The system of claim 1, wherein the programming logic further comprises logic responsive to the location query to return one of a location message or a redirect message, wherein the location server receiving the query returns the location message if the queried location server contains location information for the desired entity, and wherein the queried location server returns a redirect message if the queried location server lacks location information for the desired entity, the redirect message comprising information for finding a location server known to have location information relevant to the location query.”

Claim 4

“The system of claim 3, wherein the information for finding the location server known to have location information relevant to the location query comprises a list of location servers and a corresponding list of each of the entities having location information on the location servers.”

Claim 5

“The system of claim 1, wherein each of the location servers stores at least a portion of the location information on a persistent storage device.”

Claim 6

“The system of claim 1, wherein the location information in the location server is maintained in an indexed location store indexed by a hash table.”

Claim 7

“The system of claim 6, wherein the location query identifying the desired entity comprises a unique identifier for the desired entity, and wherein the programming logic stored on the location server further comprises programming logic for applying an index function to the

unique identifier to retrieve at least a portion of the location information associated with the unique identifier in the indexed location store.”

Claim 10

[10.pre] “A system having a plurality of location servers for managing location information and providing location information to location queries, the system comprising:”

[10.a] “a plurality of location servers containing location information corresponding to a plurality of entities, the location information formatted according to a transfer protocol configured for manipulating location information, and comprising at least one application server address, wherein the plurality of location servers are arranged in a cluster topology such that each location server contains a unique set of location information of an aggregate set of the location information; and”

[10.b] “programming logic stored on each of the plurality of location servers responsive to a location query for a desired identifier to return one of a location message, wherein a queried location server returns a location message if the queried location server contains location

information for the desired identifier, and a redirect message if the queried location server does not contain location information relevant to the desired identifier, wherein the redirect message comprises information for finding a location server having location information related to the desired identifier.”

Claim 11

“The system of claim 10, wherein the location information comprises geographic location information.”

Claim 12

“The system of claim 10, wherein the location information comprises network address information.”

Claim 13

“The system of claim 12, wherein the network address information comprises a network address of a data repository containing data pertaining to the desired identifier.”

Claim 14

[14.pre] “A method of handling location queries in a network, the network comprising a plurality of location servers, each location server containing a unique set of location information of an aggregate

set of location information correlating each of a plurality of identifiers with at least one location, the method comprising:”

[14.a] “receiving a location query from a client at one of the plurality of location servers, the location query requesting an entity’s location;”

[14.b] “sending a location response message to the client if the queried location server contains location information relevant to an entity identified in the query, the location response message comprising location information identifying at least one application server containing information relevant to the entity identified in the query; and”

[14.c] “sending a redirect message to the client if the queried location server does not contain data location information relevant to the entity identified in the query, the redirect message comprising information for finding a location server storing the entity identified in the query.”

Claim 15

“The method of claim 14, further comprising applying information in the location query from the client to an indexing function to determine if the queried location server contains location information relevant to the entity identified in the query.”

Claim 16

“The method of claim 15, wherein each of the plurality of location servers comprises an indexed table of identifiers and associated locations, and wherein applying information in the location query comprises applying an identifier in the location query to a hash function.”

Claim 17

[17.pre] “A method of scaling at least one of capacity and transaction rate capability in a location server in a system having a plurality of location servers for storing and retrieving location information, wherein each of the plurality of location servers stores unique set of location information of an aggregate set of location information, the method comprising:”

[17.a] “providing a transfer protocol configured to transport identifier and location information, the location information specifying the location of information related to the identifier;”

[17.b] “storing location information formatted according to the transfer protocol at a first location server;”

[17.c] “receiving an identifier and a location relevant to the identifier at the first location server;”

[17.d] “storing the received location in a location store at the first data location server, the location store comprising a plurality of identifiers, each identifier associated with at least one location, wherein the received location is associated with the received identifier in the location store; and”

[17.e] “transferring a portion of the identifiers and associated locations to a second data location server when a performance criterion of the first location server reaches a predetermined performance limit.”

Claim 23

“The method of claim 17, wherein the performance criterion comprises an amount of available persistent storage space in the first location server.”

Claim 24

“The method of claim 17, wherein the performance criterion comprises a transaction rate limit.”

Claim 25

“The method of claim 17, wherein the transaction rate limit comprises a processor speed limit.”

Claim 26

“The method of claim 17, wherein the transaction rate limit comprises a network connection bandwidth limit.”

Claim 30

“The method of claim 17, wherein transferring a portion of the identifiers and associated locations to a second location server when a performance criterion of the first data location server reaches a predetermined performance limit further comprises monitoring the performance criterion and automatically transferring the portion of identifiers and associated locations when the first location server reaches the predetermined limit.”

CERTIFICATE OF COMPLIANCE

Pursuant to 37 C.F.R. § 42.24(d), the undersigned certifies that the foregoing
Petition for *Inter Partes* Review of U.S. Patent No. 7,233,978 contains, as
measured by the word-processing system used to prepare this paper, 13,928 words.
This word count does not include the items excluded by 37 C.F.R. § 42.24 as not
counting towards the word limit.

Respectfully submitted,

Dated: June 28, 2024

By: /Naveen Modi/
Naveen Modi (Reg. No. 46,224)
Counsel for Petitioner

CERTIFICATE OF SERVICE

I hereby certify that on June 28, 2024, I caused a true and correct copy of the foregoing Petition for *Inter Partes* Review of U.S. Patent No. 7,233,978 and supporting exhibits to be served via express mail on the Patent Owner at the following correspondence address of record as listed on Patent Center:

Reichman Jorgensen Lehman & Feldberg LLP
100 Marine Parkway
Suite 300
Redwood City, CA 94065

A courtesy copy was also sent via electronic mail to the Patent Owner's litigation counsel at the following addresses:

Christine E Lehman (clehman@reichmanjorgensen.com)
Jaime Francisco Cardenas-Navia (jcardenas-navia@reichmanjorgensen.com)
Gina H Cremon (gcremona@reichmanjorgensen.com)
Khue Hoang (khoang@reichmanjorgensen.com)

Reichman Jorgensen Lehman & Feldberg LLP

Respectfully submitted,

By: /Naveen Modi/
Naveen Modi (Reg. No. 46,224)
Counsel for Petitioner