

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

**DECLARATION OF DR. SANDEEP CHATTERJEE
IN SUPPORT OF PETITION FOR INTER PARTES REVIEW
OF U.S. PATENT NO. 7,233,978**

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	BACKGROUND AND QUALIFICATIONS	1
III.	MATERIALS REVIEWED	6
IV.	PERSON OF ORDINARY SKILL IN THE ART AND THE TIME OF THE ALLEGED INVENTION	8
V.	TECHNICAL BACKGROUND	9
	A. Distributed Computing Systems	10
	B. Hash Functions and Applications	12
VI.	THE '978 PATENT	15
	A. Description	15
VII.	OVERVIEW OF THE PRIOR ART	17
	A. Kahn	17
	B. Vingralek	24
	C. Krasner	26
	D. Sato	29
VIII.	CLAIM CONSTRUCTION	31
	A. "identifier"	32
	B. "location information"	33
	C. "location server"	33
	D. "client"	33
	E. "plurality of location servers"	33
IX.	THE PRIOR ART DISCLOSES AND/OR SUGGESTS ALL RECITED FEATURES OF CLAIMS 1-7, 10-17, 23-26, and 30 OF THE '978 PATENT	35
	A. Kahn Discloses and/or Suggests Claims 1, 3, 5, 6, 10, and 12- 16	36

1.	Claim 1	36
2.	Claim 3	66
3.	Claim 5	70
4.	Claim 6	72
5.	Claim 10	75
6.	Claim 12	82
7.	Claim 13	84
8.	Claim 14	84
9.	Claim 15	91
10.	Claim 16	92
B.	Kahn in View of Krasner Discloses and/or Suggests Claims 2 and 11	94
1.	Claim 2	94
2.	Claim 11	98
C.	Kahn in View of Vingralek Discloses and/or Suggests Claims 3-4, 7, 10, 12-17, 23, and 30	99
1.	Claim 3	99
2.	Claim 4	111
3.	Claim 7	114
4.	Claim 10	116
5.	Claim 12	118
6.	Claim 13	118
7.	Claim 14	118
8.	Claim 15	120
9.	Claim 16	121
10.	Claim 17	122
11.	Claim 23	135
12.	Claim 30	136

D.	Kahn in View of Vingralek and Krasner Discloses and/or Suggests Claim 11	137
1.	Claim 11	137
E.	Kahn in View of Vingralek and Sato Discloses and/or Suggests Claims 24-26	138
1.	Claim 24	138
2.	Claim 25	142
3.	Claim 26	142
X.	CONCLUSION.....	145

I, Sandeep Chatterjee, declare as follows:

I. INTRODUCTION

1. I have been retained as an independent expert consultant in this proceeding before the United States Patent and Trademark Office (“PTO”) regarding U.S. Patent No. 7,233,978 (“the ’978 patent”) (Ex. 1001).¹ I have been asked to consider whether prior art references disclose or suggest the features recited in claims 1-7, 10-17, 23-26, and 30 (“the challenged claims”) of the ’978 patent. My opinions are set forth below.

II. BACKGROUND AND QUALIFICATIONS

2. My qualifications are stated more fully in my curriculum vitae, which is attached as Ex. 1003. Below is a summary of my education, work experience, and other qualifications.

3. I am the Chief Executive Officer of Experantis LLC (“Experantis”), a technology consulting company. Previously, I was the Co-founder, Executive Vice President and Chief Technology Officer of SourceTrace Systems, Inc., a

¹ In this Declaration, I refer to exhibits that I understand are to be attached to the petition for *Inter Partes* Review of the ’978 patent.

technology and services company enabling the delivery of secure remote electronic services over landline and wireless telecommunications networks.

4. I received my Bachelor's degree in Electrical Engineering and Computer Science from the University of California, Berkeley in 1995. I received my Master's degree in Computer Science from the Massachusetts Institute of Technology (MIT) in 1997, and my Doctorate in Computer Science from MIT in 2001. I received a certificate of completion for an executive education program on global leadership from Harvard University in 2011.

5. My doctoral dissertation at MIT, entitled "Composable System Resources for Networked Systems," involving networked and distributed computer systems and architectures, was selected as one of the top inventions in the history of MIT's Laboratory for Computer Science. This invention is showcased in a time capsule at the Museum of Science in Boston, Massachusetts. Other recipients of this honor include Bill Gates, the founder of Microsoft, and Tim Berners-Lee, the inventor of the World Wide Web.

6. As part of applications of my doctoral research, I developed hardware and software systems for intelligent environments within homes and offices. Some of these devices included televisions, digital picture frames, refrigerators and

children's toys. The distributed computing system of these intelligent environments included distributed data storage and retrieval.

7. In 2011, I was named a Young Global Leader. This honor, bestowed each year by the World Economic Forum, recognizes and acknowledges the top leaders—all below the age of 40—from around the world for their professional accomplishments, commitment to society, and potential to contribute to shaping the future of the world. In 2016, I was appointed to the World Economic Forum's expert network as an expert in technology and innovation.

8. From 1997, I was the Entrepreneur-in-Residence at FidelityCAPITAL, the venture capital arm of Fidelity Investments. In 1999, I founded and served as President and Chief Technology Officer (CTO) of Satora Networks, which developed tools and technologies for building appliances and services for the Internet using wireless and other technologies to extend it beyond the desktop.

9. In 2001, I joined Bluestone Software's Mobile Middleware Labs as a Senior Engineer developing applications and systems infrastructure for enterprise Java/J2EE, Web services, and enterprise mobile solutions. After the completion of Hewlett-Packard's (HP) acquisition of Bluestone, I became a Senior Member of the Technical Staff at HP's Middleware Division. I was responsible for

architecting and developing the company's next-generation Web services platform for enterprise as well as mobile environments, known as the Web Services Mediator.

10. I was part of the Expert Group that developed the JSR-00172 J2ME (Java 2 Platform, Micro Edition) Web Services Specification, the worldwide industry standard for mobile Web services. I am the co-author, with James Webber, of the book "Developing Enterprise Web Services: An Architect's Guide" (published by Prentice-Hall in 2004). This book has been adopted by over 100 universities and colleges around the world, and has been translated or reprinted in a number of countries around the world.

11. I have significant experience in developing complex computing systems. For example, through a contract between HP and the United States Agency for International Development (USAID), I architected and led the development of one of the first mobile banking solutions. This system enabled customers to use their mobile phones and other wireless handsets to connect with the core banking systems of banks and other financial institutions, and perform transactions without having to travel to bank branches. This system supported many banking transactions, including deposits, withdrawals, loan applications, loan disbursements and loan repayments.

12. Later, after SourceTrace Systems' acquisition of this technology, I led the expansion of this solution into multiple countries and into multiple industries. Banks and other financial services companies utilized this technology to make their tellers more efficient, to provide self-service ATMs within branches, and to provide remote access to banking services. Additionally, through our licensing agreement with Telefonica, one of the largest cellular and telecommunications companies in the world, this solution was deployed in various other industries, including logistics and asset management and customer relationship management. Bloomberg Television selected and featured this technology and the company I co-founded to commercialize this technology on Bloomberg TV's "Bloomberg Innovators" program.

13. I have been a retained expert witness for various disputes that involved significant technology issues, and I have been qualified as a technology expert by U.S. District and State Courts, including in California, Delaware, Florida and Texas in technology areas that are relevant to this case, including but not limited to: computer software systems, distributed computing systems, and mobile and wireless systems. I have previously testified through declaration or expert report, at deposition, and at trial in numerous intellectual property and commercial litigation matters, including for patent litigation, copyright and trade secret

misappropriation litigation, and contract dispute cases. I have submitted more than one hundred and fifty expert declarations and expert reports, testified at deposition more than eighty times, and testified at trial or at hearings at least twelve times. I have been identified as one of the top 1000 patent professionals in the world, and I am listed in the IAM Patent 1000, which identifies the world's leading patent litigation and prosecution attorneys, as well as damages and technology expert witnesses.

14. Experantis is being compensated for my time on this matter at my standard hourly rate and reimbursed for any expenses that I incur related to my work in this matter. Neither Experantis nor I have any financial interest in the outcome of this matter, and Experantis will be paid for my time regardless of the outcome of this matter.

III. MATERIALS REVIEWED

15. The opinions contained in this Declaration are based on the documents I reviewed, my professional judgment, as well as my education, experience, and knowledge regarding systems and processes in the field of distributed computing systems.

16. In forming my opinions expressed in this Declaration, I reviewed the following materials:

Declaration of Dr. Sandeep Chatterjee
Patent No. 7,233,978

Ex. 1001	U.S. Patent No. 7,233,978 to Overton et al. (“the ’978 patent”)
Ex. 1003	Curriculum Vitae of Sandeep Chatterjee, Ph.D.
Ex. 1004	Prosecution History of U.S. Patent No. 7,233,978
Ex. 1005	WIPO Application No. 1997/043717 to Kahn et al.
Ex. 1006	U.S. Patent No. 6,473,781 to Skagerwall et al.
Ex. 1007	R. Vingralek, <i>Distributed file organization with scalable cost/performance</i> (1993)
Ex. 1009	U.S. Patent No. 6,104,338 to Krasner
Ex. 1011	Japanese Patent Application Publication No. H09-231180 to Sato et al.
Ex. 1014	RFC 768, User Datagram Protocol (Aug. 28, 1980)
Ex. 1016	U.S. Patent No. 5,764,756
Ex. 1056	Joseph B. Greene, <i>Oracle8 Server Unleashed</i> (1998)
Ex. 1057	Joseph B. Greene, <i>Oracle DBA Survival Guide</i> (1995)
Ex. 1058	Betty Salzberg, <i>File Structures</i> (1988)
Ex. 1059	Andrew P. Black & Yeshayahu Artsy, <i>Implementing Location Independent Invocation</i> (1989)

I also considered any other documents and materials I refer to in this Declaration.

17. My opinions contained in this declaration are based on the documents I reviewed and my knowledge and professional judgment. My opinions have also been guided by my appreciation of how a person of ordinary skill in the art would

have understood the state of the art, the prior art, and the claims and the specification of the '978 patent at the time of the alleged invention, which I discuss below.

18. I have been asked to consider that the time of the alleged invention of the '978 patent was around 1998 (including and up to July 8, 1998), which corresponds to the filing date of the earliest patent application associated with the '978 patent.

19. Based on my experience and expertise, it is my opinion that certain references disclose and/or suggest all the features recited in challenged claims of the '978 patent, as I discuss in detail below.

IV. PERSON OF ORDINARY SKILL IN THE ART AND THE TIME OF THE ALLEGED INVENTION

20. I am familiar with the level of ordinary skill in the art regarding the '978 patent as of around July 8, 1998. Considering the '978 patent, the technology, the educational level and experience of workers in the field relating to the patent, and problems and solutions in that field, and drawing on my own experience, I believe a person of ordinary skill in the art at the time of the alleged invention (around July 8, 1998) 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.

21. My opinions in this Declaration regarding the '978 patent and the prior art (including the state of the art) are from the perspective of one of ordinary skill in the art as I defined above, during the relevant timeframe (e.g., the time of the alleged invention), which I discussed above as being around July 8, 1998.

V. TECHNICAL BACKGROUND

22. Below, I present a brief overview of certain aspects of systems and processes relating to distributed computing systems at the time of the alleged invention for the '978 patent. The functionalities and concepts I describe below in this technical background section reflect the state of the art that a person of ordinary skill in the art would have had knowledge of and understood prior to and at the time of the alleged invention of the '978 patent. I rely on, and incorporate as applicable (even if not expressly mentioned below in Section IX), the following disclosures and opinions to support my opinions in this Declaration, including those opinions relating to how the prior art discloses and/or suggests the challenged claims of the '978 and how and why a person of ordinary skill in the art would have been motivated to consider and combine the disclosures and suggestions from that prior art as I explain below in Section IX. The concepts

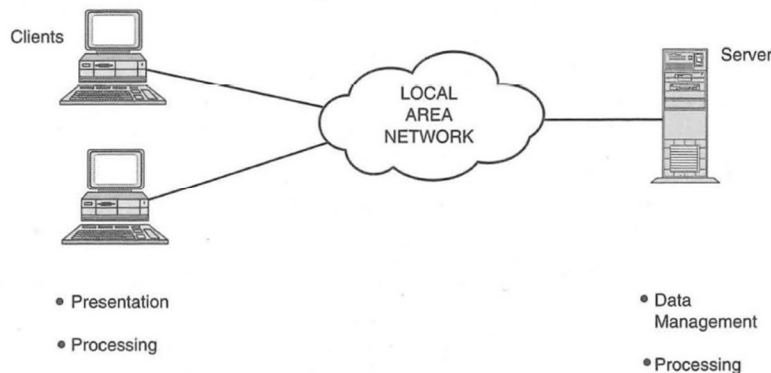
below would have been within the knowledge and mindset of a person of ordinary skill in the art at the time.

A. Distributed Computing Systems

23. A distributed network is a network where data is distributed instead of stored centrally. (Ex. 1005, 73:27-31 (“This strategy assures a random distribution of handle server requests among many handle servers distributed on a network without a central nodal point in the system (for reliability).”); *see also* Ex. 1006, 8:15-19 (“It is a key aspect of the invention that the data base of object data is distributed over a plurality of directory servers DS; DS; DS1-DSn, allowing that the number of objects can be scaled almost arbitrarily by adding new directory servers to the existing ones for storing new object data.”).) Within the distributed network are multiple databases which are connected to operate as a singular database. (Ex. 1056, 164 (“The ideal form, where a large network of connected databases act as one database, replicating one another’s data, is a *distributed database*. A distributed database is singular in that although data physically exists at many sites, it is still one database at the highest level that is simply distributed physically....”); Ex. 1001, 12:46-49 (disclosing that hash functions were well known at the time of the alleged invention).) “In distributed databases, you have multiple, completely equal databases,” where the distributed databases may use a

client-server-architecture to accomplish tasks. (Ex. 1057, 680.) As shown with reference to Figure 46.1 below, a client-server architecture in its most simple form includes “two intelligent devices (your desktop PC and the server) that share the work load between them.” (Ex. 1057, 674.)

*Figure 46.1.
Client-server
architecture.*



Alternatively, there may be a network environment that involves a client, a database server, and an application server (which handles most of the application processing). (*Id.*)

24. Distributed network application servers can take applications and provide “reliable, scalable, and robust applications services.” (Ex. 1056, 36 (“Network computing is about universal access to data...universal access to applications...network services...and bringing new applications online faster and at a lower cost.... Powerful applications servers will take simple objects such as Enterprise JavaBeans and turn them into reliable, scalable, and robust applications services.”).) In particular within a distributed network, “load distribution and load

balancing are basic means of improving performance.” (Ex. 1011, ¶[0003]
 (“Furthermore, load distribution and load balancing are basic means of improving
 performance in distributed systems.”).)

B. Hash Functions and Applications

25. A hash function takes an input, usually in the form of a key, and
 calculates the location of information (e.g., an address) associated with that key.
 (Ex. 1058, 202 (“Suppose that we knew that the keys of the records of some file
 were the numbers from 1 to 1000. Then we could store the record with key 1 in
 position 1 on the disk, the record with key 2 in position 2, and so on. Then to find a
 record with key x, we could go directly to position x. The value of the key would
 yield the address of the key. This would be faster than using a B+-tree for
 individual record searches. This would be a one-disk-access method. Hashing is a
 generalization of this process.”).) For example, “[a] *hashing algorithm* or *hash
 function* is a means of calculating the disk address of a block containing a given
 record from the value of its key.” (*Id.*)

26. “The point of hash functions is to take a highly nonuniform
 distribution—probably one with data unevenly clustered—and transform it into
 one which is nearly uniform.” (*Id.*, 203.) For example, if data is stored non-
 linearly, a hash function is designed to be “fast for fetching, inserting, and

deleting” files within a system. (*Id.*, 215.) Hash functions are also optimal “[w]hen a large number of individual records must be fetched.” (*Id.*, 241 (“Hashing is the best file structure for single-record fetches. **When a large number of individual records must be fetched**, as with automatic teller machines or airline reservations, **this is the best method.**”); *see also id.*, 215 (“Usually, right after a hash table has been created, fetching takes about one disk access. This is the method of choice for many files and many database management systems. **When most of the operations on the file are individual-record fetches, hashing is the optimal file structure.**”).)

27. Hash functions are used in many applications, but as an example, a hash function can be used to locate a tax record associated with a social security number (SSN) within a distributed system, where the SSNs are not stored numerically. (*Id.*, 209 (“For example, suppose that the California tax records are organized by a hash table based on social security number. This means that there is a randomizing function calculated from the social security number and a remainder taken from this number. The records are *not* stored in order of social security number. Two records whose social security numbers differ only in the last digit may be nowhere near each other.”).) To locate a tax record associated with a SSN, a hash function is applied to the SSN. (*Id.*) The result of the hash function is then

used to fetch the tax record. (*Id.*, 209-210 (“If we want to read the records of all the California taxpayers whose social security number is between 000-00-0000 and 000-99-9999 from the file of California tax records, we...have to make 1 million separate fetches for the 1 million different numbers between 0 and 999,999..... Making one million hash-table fetches will take about 7 hours.”), 242 (“For creating a hash table from an already existing large file, as happens initially for the linear-hashing table or repeatedly for the traditional hashing method, we need to sort the records by their hash value and create the table in the last pass.”).)

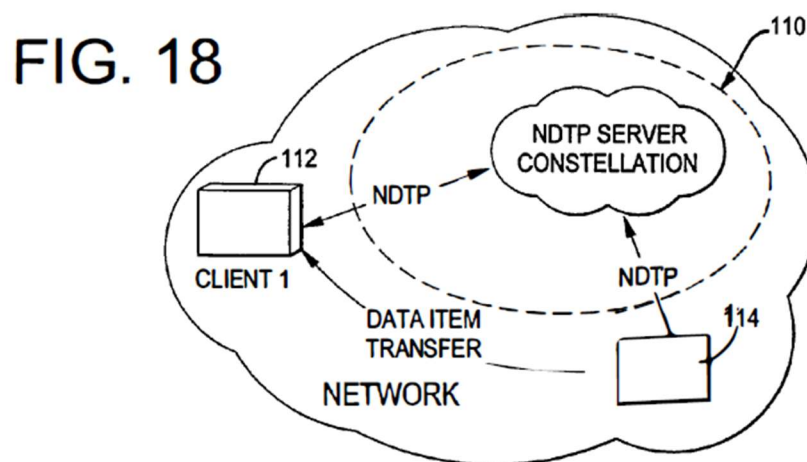
28. When data is not found at the specified location, perhaps due to the data being stored in a new location in the system, a message with a forwarding address may be returned to the requesting system. (Ex. 1059, 553-54 (“If object *o* is at node *n* after its a^{th} move, then the pair $\langle n, a \rangle$ is a *temporal address descriptor* (*tad*) for *o*.... If *o* later migrates to *m*, its descriptor on *n* is updated to include new *tad* $\langle m, 1 \rangle$. Assuming that objects do not migrate too often, this *tad* represents a forwarding address that is likely to remain correct for some time.... **If *o* is no longer at *m*, then *m* will normally have a newer *tad* for *o*. What should *m* do? There are two alternatives: (1) return a failure indication along with its stashed *tad*, or (2) propagate the invocation to the location indicated in its *tad*.** In Hermes, we chose a hybrid of the two approaches. If the invoked node does not

have the object, the invocation is propagated to the location indicated in the *tad*; this continues up to some small number of hops..... **Otherwise, a failure with a newer *tad* is returned to the originator of the search, *n* in this example, which starts afresh with the new forwarding pointer.”.)** For example, if the tax record is no longer stored at the original location, a message can be returned to the requesting system with the updated location. (*Id.*)

VI. THE '978 PATENT

A. Description

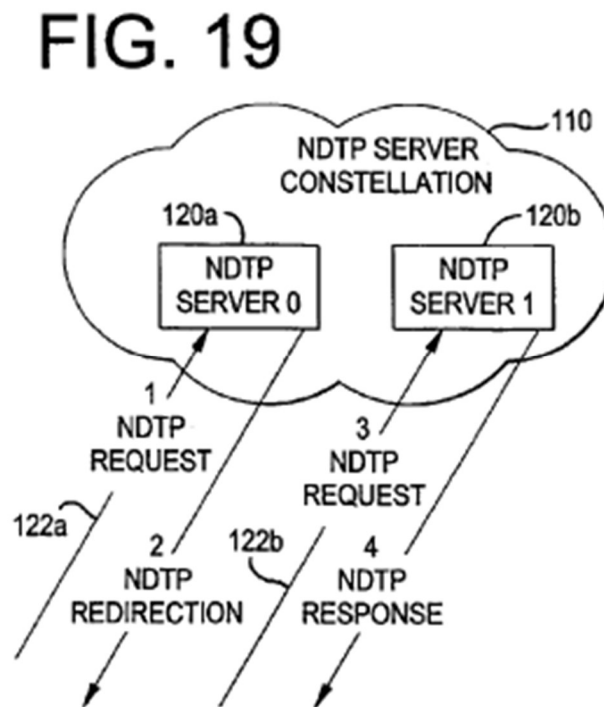
29. The '978 patent describes storing and retrieving information across a network (Ex. 1001, 1:30-33, 1:56-58) and a network distributed tracking protocol (NDTP) (*id.*, 2:19-21, 7:55-61). Figure 18 shows an NDTP server constellation where a client consults servers that return strings in response to a client request. (*Id.*, 18:50-58.)



(*Id.*, FIG. 18.)

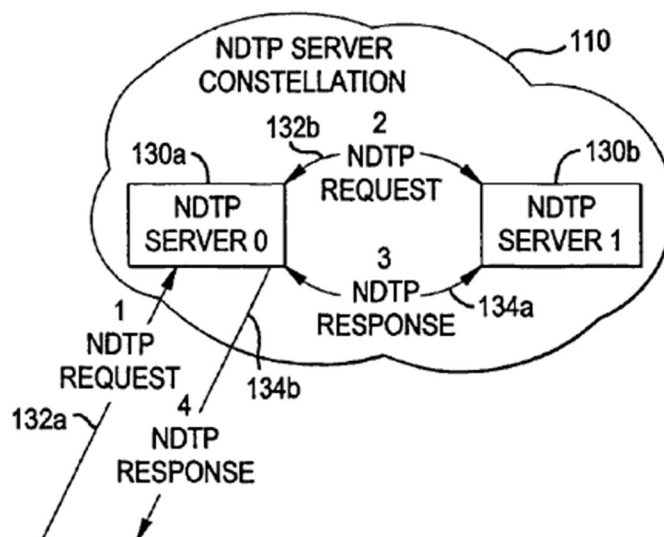
30. Server constellations have two organizational approaches: client-centric (shown in Figure 19) and server-centric (shown in Figure 20). In the client-centric approach, processing emphasis is toward clients rather than toward servers.

(*Id.*, 18:60-19:3.)



(*Id.*, FIG. 19.) In contrast, in the server-centric approach, server processing is emphasized. (Ex. 1001, 19:20-24.)

FIG. 20



(*Id.*, FIG. 20.)

31. In both the client-centric and server-centric approaches, a client transmits an identifier specifying an entity to a server in a client request. (Ex. 1001, 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.*)

VII. OVERVIEW OF THE PRIOR ART

A. Kahn

32. Kahn discloses a method for managing digital objects stored in one or more repositories in a network, where the digital objects are located by one or

more pointers associated with a handle. (Ex. 1005, Abstract (“A method of managing digital objects (1004) in a network is presented. The objects are stored at [repository] 1002.... Other aspects include providing multiple servers (1042) each of which accepts identifiers of a subset of digital objects and returns corresponding pointers to the locations of the digital objects in the network.”), 6:24 (“There may be multiple repositories.”).) Kahn explains that multiple handle servers in the network each may accept handles of the digital objects and return corresponding pointers that indicate the digital objects’ location in the network. (*Id.*, Abstract, 6:28-35 (“A service, accessible on the network, is provided for locating the handle associated with a digital object. The handle is used to obtain a pointer to the network location of...the digital object.”), 7:5-20 (“A handle server, accessible on the network, provides the pointer in response to the presentation of a handle. Multiple servers provide the service, each serving a portion of the handle space. []). Hash values are computed on the handles and the hash values are distributed among multiple handle servers, each handle server having a table which associates handles with pointers.”), 21:6-24 (“[M]ultiple servers...provide...handle-to-pointer translation services (called ‘handle servers’)....[. It should also be clear that services can be provided on a widely distributed basis by multiple similar servers at different locations on the network, or by single centralized servers.”).)

33. Kahn discloses with reference to Figure 1 a high-level diagram of an exemplary system that includes a repository 1002 (highlighted in blue) for storing digital objects 1004 (highlighted in green) in the network, where the digital objects are identified by their handle 1012 (highlighted in yellow) and associated pointers. (*Id.*, Abstract (“A method of managing digital objects (1004) in a network is presented. The objects are stored at [repository] 1002.... Other aspects include providing multiple servers (1042) each of which accepts [handles (1012)] of a subset of digital objects and returns corresponding pointers to the locations of the digital objects in the network.”).) Handle-pointer mappings are stored in tables within handle servers 1042 (highlighted in red). (*Id.*, Abstract, 7:17-20 (“...each handle server having a table which associates handles with pointers.”), 11:16-18 (“A ‘repository’ 1002 is a digital storage system into which digital objects 1004 may be placed and retained for possible subsequent retrieval.”), 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles....”).)

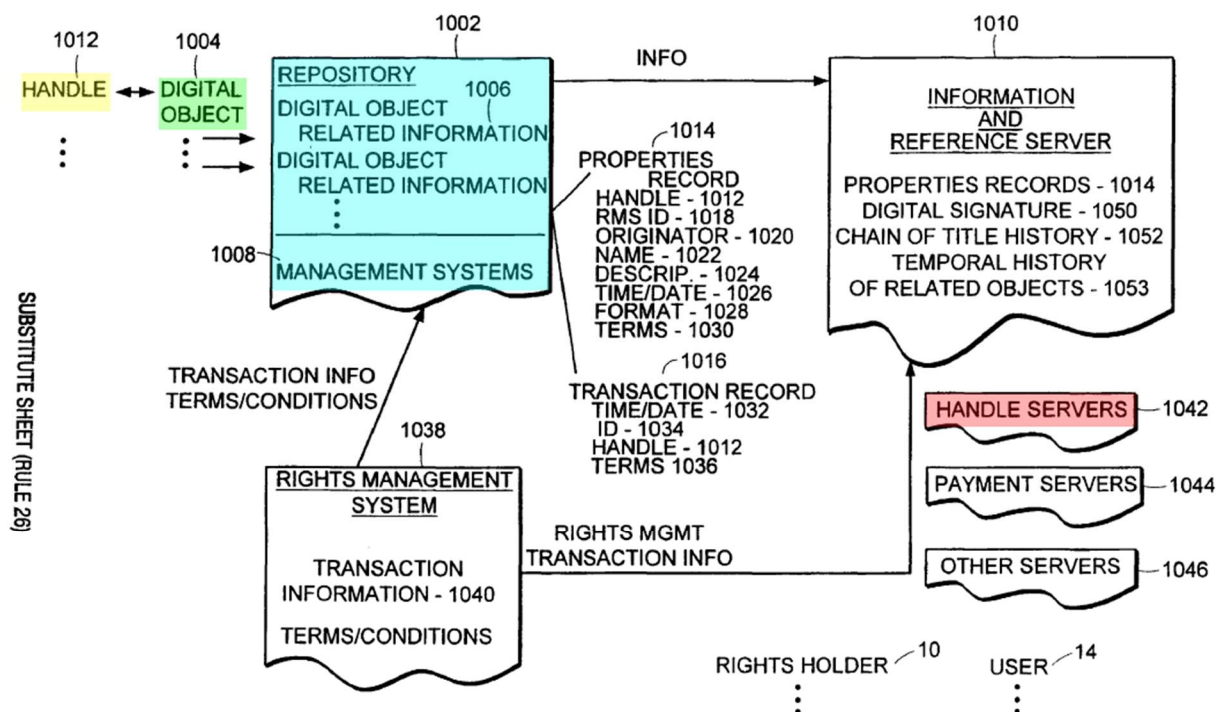


FIG. 1

(*Id.*, FIG. 1 (annotated).)

34. Kahn explains that a digital object may include conventional digital representations of works (e.g., books, papers, images, sounds, software) and each digital object is associated with a unique handle used to identify the digital object. (*Id.*, 1:16-20 (“Thus digital objects may include conventional representations of works (books, papers, images, sounds, software), and more broadly any digital material which is capable of producing desired manifestations for a computer user.”), 21:3-5 (“Each digital object has a ‘handle’, a concise unique identifier for a digital object used for storage and retrieval operations and other repository

functions.”.) Kahn discloses that a digital object is stored in a repository in a distributed network, as “[digital] objects are accessed at the repository via the network,” where the network does not have “a central nodal point in the system (for reliability).” (*Id.*, Abstract (“The [digital] objects are stored at locations...accessible in the network. []. Other aspects include providing multiple servers [] each of which accepts identifiers of a subset of the digital objects and returns corresponding pointers to the locations of the digital objects in the network....”), 2:25-30 (“Making digital objects available on networks (e.g., Internet), gives rise to at least four specific activities of concern. The first is the ease of movement of digital objects already contained in a computer network environment....”), 3:15-19 (“[T]he [digital] objects are stored at locations accessible in the network”), 6:9-35 (“Copies of the digital objects are stored in a repository.... []. The objects are accessed at the repository via the network.... []. The handle is used to obtain a pointer to the network location of...the digital object.”), 11:16-18 (“A ‘repository’ 1002 is a digital storage system into which digital objects 1004 may be placed and retained for possible subsequent retrieval.”), 21:21-24 (“...services can be provided on a widely distributed basis by multiple similar servers at different locations on the network....”), 73:23-31 (“One or more pointers, once returned, identifies the network location of the one or more

repositories (if one is associated with the object)....This strategy assures a random distribution of handle server requests among many handle servers distributed on a network without a central nodal point in the system (for reliability).”) For example, Kahn discloses that digital objects are stored on the Internet. (*Id.*, 2:25-30 (“Making digital objects available on networks (e.g., Internet), gives rise to...activities of concern. The first is the ease of movement of digital objects already contained in a computer network environment allowing the creation of multiple copies in multiple machines in fractions of second.”).)

35. Kahn discloses further details explaining that the one or more pointers returned by a handle server indicate the location of a stored digital object in the distributed network. (*Id.*, Abstract (“[M]ultiple servers (1042) each of which accepts identifiers of a subset of the digital objects and returns corresponding pointers to the locations of the digital objects in the network....”), 3:19-22 (“[A] pointer indicating the location of the stored digital object is also stored in the network.”), 6:30-35 (“The handle is used to obtain a pointer to the network location of an accessible copy [of the digital object].”), 28:32-34 (“Object pointers contain location information for locating digital objects [].”) For example, Figure 8 describes a client system retrieving one or more pointers for digital objects. (*Id.*, FIG. 8, 46:25-47:25 (“Given a handle, the following steps, shown in Figure 8, are

followed to obtain a set of pointers associated with the handle.”); *see also id.*,
72:32-74:32.)

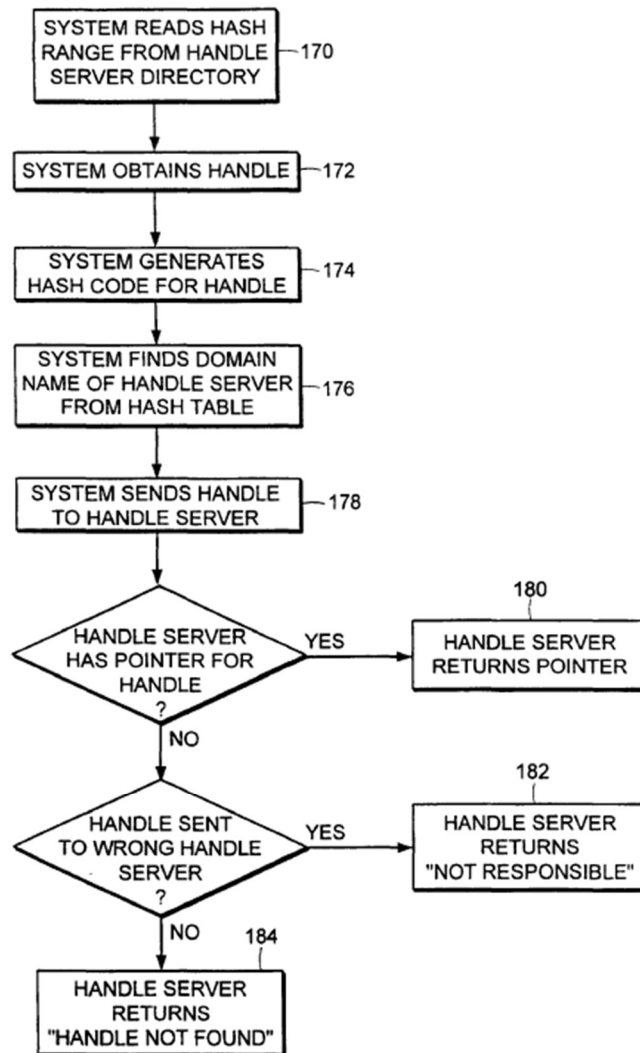


FIG. 8
SUBSTITUTE SHEET (RULE 26)

(*Id.*, FIG. 8.)

B. Vingralek

36. Vingralek discloses a “distributed file organization system” where a file is “organized into buckets that are spread across multiple servers, where a server may hold multiple buckets.” (Ex. 1007, 4.) Vingralek further explains that clients can modify the file’s distribution. (*Id.*, 9 (“Consider a file consisting of keyed records that is spread across a number of servers connected by a network. []). A server keeps records of the file primarily on disk and we assume that servers are dedicated to managing a distributed file. We assume that there is a distributed group of clients that issue requests to insert, delete, and retrieve records for a given key *K* in the file. Consequently, the file grows or shrinks dynamically.”).) The approach to file distribution in Vingralek is hash-based as hashing is used to distribute keys across buckets, where keys are used to retrieve part of a particular file. (*Id.*, 5 (“The clients locate servers for insertion, update, deletion, and key-based retrieval through a hash function....”), 6 (“This paper presents a new approach for hash-based distributed files....”), 9 (“Thus, a hash file organization is the most appropriate in such an environment.”).)

37. Additionally, Vingralek teaches changes in the data/file distribution may “result [in] changes to the server-locating hash function.” (*Id.*, 5.) In that scenario, the client is notified “in order to update the client’s stale information on

the hash function.” (*Id.*; *see also id.*, 5 (“To limit the communication overhead, clients learn about bucket splits and the resulting changes to the server-locating hash function only when a client request is directed to the ‘wrong’ server.”).) For example, each client and server may have stale information on a parameter called the “file level,” which is the highest server level among all servers. (*Id.*, 10.) Thus, the hash function, which depends on the “file level” parameter, may also be stale if there has been a change to the servers. (*Id.*)

38. Moreover, instead of updating a client’s hash function immediately after a change to the data/file distribution has occurred, Vingralek discloses such an update is not provided to the client until “a client request is directed to the ‘wrong’ server” in order to limit or reduce the “communication overhead.” (*Id.*, 5 (“To limit the communication overhead, clients learn about bucket splits and the resulting changes to the server-locating hash function only when a client request is directed to the ‘wrong’ server. In such a case, the server forwards the client’s request to the proper server, and notifies the client in order to update the client’s stale information on the hash function. The most salient feature of this approach is that its communication overhead is largely independent of the number of servers and clients in the system; thus, it is considered a scalable approach.”); *see also id.*, 6 (providing “updates...to client...in a deferred manner”).) Indeed, Vingralek

explains that this approach would have decoupled the communication overhead from the number of servers/clients in the system and improved scalability. (*Id.*, 5 (“The most salient feature of this approach is that its communication overhead is largely independent of the number of servers and clients in the system; thus, it is considered a scalable approach.”).)

C. Krasner

39. Krasner relates to a wireless communication system in a client-server architecture. For example, Krasner discloses a wireless communication system that has location servers that store location information pertaining to a mobile GPS unit. (Ex. 1009, Abstract (“A method and an apparatus for operating a satellite positioning system (SPS) receiver in a client/server architecture.”), 4:10-15 (“The system...includes a location server 25 which communicates through a wireless communication system with the communication system which is coupled to or part of the mobile GPS unit 12. The basestation 25 typically includes storage 26 for storing a time sequence of differential GPS and satellite ephemeris information.”).)

40. In particular, Krasner discloses a “system for tracking the location of a mobile GPS receiver over time as it moves.” (*Id.*, 3:56-57.) Krasner describes its system as an improvement upon similar prior art systems which allow for tracking the position of a mobile GPS unit (e.g., the GPS receiver of a cellular telephone or

a car's mobile telephone) over time. (*Id.*, 1:48-51 (“Recently, GPS receivers have been used with radio transmitters, such as a cellular telephone or a mobile telephone in a car to transmit the position of the receiver as it moves.”), 2:39-41 (“[I]t is desirable to provide an improved method and system for providing multiple sets of position information over a period of time through a mobile GPS unit.”).) Krasner explains that the mobile GPS unit receives GPS signals at various positions and determines pseudoranges from those signals, which it stores along with a corresponding time stamp in its own memory. (*Id.*, 5:49-53.) Krasner's system also includes a location server (also called a basestation server) which is in wireless communication with the mobile GPS unit that is being tracked. (*Id.*, 4:10-13.) Krasner explains that when a predetermined type event occurs, the mobile GPS unit may transmit the pseudoranges and corresponding time stamps to the location server via the wireless communication system. (*Id.*, 4:66-5:24 (“When the predetermined type event does occur,...the stored pseudoranges and the corresponding time stamps are transmitted via a wireless communication system...to the location server.”).) Examples of such predetermined type events include the mobile GPS unit having collected a certain number of signals (e.g., transfer to the location server occurs after a mobile GPS unit has collected seven sets of signals); the mobile GPS unit reaches a memory limit; a predetermined

period of time has lapsed since the last transmission; or an emergency event is detected, such as a car accident, inflated airbag, or other alarm condition. (*Id.*, 5:49-6:15.)

41. After the pseudoranges and time stamps are transmitted to a location server, the mobile GPS unit may clear that portion of its memory to allow for further collection of pseudoranges and corresponding time stamps. (*Id.*, 5:20-24 (“[T]he memory which stored the pseudoranges and the time stamps is cleared for that portion of the memory. This will permit another set of pseudoranges to be collected along with their corresponding time stamps and stored and later transmitted.”).) Krasner explains that the location server may “determine[] a position of the transmitting mobile GPS unit may apply a differential GPS correction (e.g., based on data from a GPS reference receiver) to the pseudoranges, and then “determine[] a position of the transmitting mobile GPs unit from each set of corrected pseudoranges and the corresponding time stamp” so that “the location server may be able to track the movement of [the] mobile unit in space and in time.” (*Id.*, 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.) Krasner discloses a wireless

communication system in which a location server stores geographical location information of a mobile GPS unit that may be provided to a user.

42. Krasner discloses that its method provides the benefits of prior approaches, e.g., by “provid[ing] a way to track the route of a moving GPS receiver” (*id.*, 2:10-12), but offers a number of advantages over prior art systems. (*Id.*, 2:10-41, 5:25-48.) For example, Krasner’s method offers “a number of advantages over the prior art technique of [the mobile GPS unit] determining a position at each point and then transmitting these positions” to the location server because unlike the prior art technique, Krasner’s method does not require an adequate view of the sky or an ability for the mobile unit to read satellite ephemeris data. (*Id.*, 5:25-48; *see also id.*, 2:10-30.) Krasner’s method is also preferable to prior art methods in which the GPS transmits pseudoranges to a location server one at a time because it would offer battery life benefits and reduce congestion in the wireless communication link. (*Id.*, 2:31-38.)

D. Sato

43. Sato relates to a method of partitioning servers in a distributed environment. For example, Sato describes a “client-server system built on a distributed system” in which a client “issue[s] a service request to a specific server and receive[s] a service.” (Ex. 1011, ¶[0003].) Sato explains that the system uses

another set of servers, i.e., service location servers, which provide location information about the location of the specific server that provide services to the client. (*Id.*, ¶[0003] (“One method to efficiently investigate the existence of a server that provides the service requested by a client and to efficiently obtain information for establishing the communicating means with the server is to use a server that services location information of various servers (hereinafter referred to as a service location server).”).) Sato further describes “the location information of each server is one of the global system resources,” and discloses a “server partitioning method that can partition the service range and manage location information in a distributed manner when the load on the service location server exceeds a predetermined threshold,” rather than having to pre-design how to manage the location information on each service location server. (*Id.*, ¶[0005].)

44. Sato explains that each “service location server” includes a “load measuring means” that measures the service load status of the server by measuring various “service load information.” (*Id.*, ¶[0009].) The measured “service load information...includes [1] the maximum number of service executions per unit time...after starting the service of the server..., [2] reference time...for determining whether or not the measurement time has passed, [3] a counter...for the number of services executed by the service location server since the reference

time...was set, and [4] a measurement time interval.” (*Id.*, ¶[0012].) Sato discloses that “[t]he maximum number of service executions...is set by the load measuring means.” (*Id.*) Sato explains that the system implements “partition criteria information...[which] has a predetermined threshold value...for the maximum number of service executions per unit time.” (*Id.*, ¶[0013].) “Whether or not to partition the server...is determined by comparing the maximum number of service executions...with the threshold value.” (*Id.*) For example, “[i]f the value of the maximum number of service executions...is greater than the threshold value..., it is determined that partitioning is necessary, and otherwise, it is determined that partitioning is not necessary.” (*Id.*, ¶[0029].) As Sato explains, if it is determined that partitioning is necessary, a server partition processing step is then executed in which information is migrated to another server. (*Id.*, ¶¶[0029]-[0030].)

VIII. CLAIM CONSTRUCTION

45. I have been asked to consider and apply in my analysis certain constructions of claim terms, as discussed below. For the remaining claim terms in the challenged claims, I have been asked to give and apply in my analysis their plain and ordinary meaning, as would have been understood by a person of ordinary skill in the art, at the time of the alleged invention (as I mentioned above

is the mid-1990s, up to and including July 8, 1998) having taken into consideration the language of the claims, the specification, and the prosecution history of the '978 patent. I have also been asked to consider and apply in my analysis the interpretations of certain terms that I understand are from a prior IPR proceeding and a district court proceeding, neither of which involves Petitioner. I have applied these understandings in my analysis and in forming my opinions where appropriate in this Declaration.²

A. “identifier”

46. I have been asked to interpret the claimed “identifier” recited in claims of the '978 patent to mean:

“a unique encoding that identifies an individual entity, and with which zero or more data location specifiers are associated in a location server.”

² As I explain below, I have been asked to interpret limitations in claim elements in particular ways, and I applied those interpretations in my analysis. (*See* below in Section IX.)

B. “location information”

47. I have been asked to interpret the claimed “location information”

recited in claims of the '978 patent to mean:

“one or more identifiers and their associated locations”

C. “location server”

48. I have been asked to interpret the claimed “location server” recited in claims of the '978 patent to mean:

“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.”

D. “client”

49. I have been asked to interpret the claimed “client” recited in claims of the '978 patent to mean:

“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.”

E. “plurality of location servers”

50. I have been asked to interpret the claimed “plurality of location servers” and related term “plurality of data location servers” recited in claims of the '978 patent to mean:

“location servers in non-hierarchical structures.”

51. As I have indicated, I applied each of the above interpretations for the terms identified above in my analysis where appropriate. These terms and their corresponding interpretations are listed below. I refer to each interpretation as “Interpretation from Other Proceedings.”

'978 Patent Terms	Interpretation from Other Proceedings
“identifier”	“a unique encoding that identifies an individual entity, and with which zero or more data location specifiers are associated in a location server”
“location information”	“one or more identifiers and their associated locations”
“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”
“client”	“a network-attached component (which may be software or hardware) that

'978 Patent Terms	Interpretation from Other Proceedings
	initiates update or lookup of identifier/location mappings from a location server with location request messages”
“plurality of location servers” ³	“location servers in non-hierarchical structures”

IX. THE PRIOR ART DISCLOSES AND/OR SUGGESTS ALL RECITED FEATURES OF CLAIMS 1-7, 10-17, 23-26, and 30 OF THE '978 PATENT

52. In my opinion, the prior art discussed below discloses and/or suggests the limitations of claims 1-7, 10-17, 23, and 30.⁴

³ I also apply this interpretation to claims reciting “plurality of data location servers.”

⁴ In this section (Section IX), I refer to exhibits other than those identified prior art reference(s) that disclose and/or suggest the claimed limitations. Such exhibits reflect the state of the art known to a person of ordinary skill in the art at the time of the alleged invention.

A. Kahn Discloses and/or Suggests Claims 1, 3, 5, 6, 10, and 12-16

1. Claim 1

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

53. I understand that this limitation is a preamble of the claim and I have been asked to assume that the preamble of claim 1 is limiting. Under such an assumption, Kahn discloses and/or suggests the limitations therein. 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. (Ex. 1005, FIG. 1, Abstract (“A method of managing digital objects (1004) in a network is presented. The objects are stored at [repository] 1002....Other aspects include providing multiple servers (1042) each of which accepts [handles (1012)] of a subset of digital objects and returns corresponding pointers to the locations of the digital objects in the network.”), 1:10-12 (“By a **‘digital object’** we broadly mean any set of sequences of bits or digits and **an associated unique identifier which we call a ‘handle.’**”), 6:9-10 (“Copies of the digital objects are stored in a repository.”), 6:24 (“There may be multiple repositories.”), 11:16-18 (“A ‘repository’ 1002 is a digital storage system into which digital objects 1004 may be placed and retained for possible

subsequent retrieval.”), 21:3-5 (“Each digital object has a **‘handle’**, a **concise unique identifier for a digital object** used for storage and retrieval operations and other repository functions.”).) A digital object may include digital representations of works (e.g., books, software). (*Id.*, 1:16-20 (“Thus digital objects may include conventional representations of works (books, papers, images, sounds, software), and more broadly any digital material which is capable of producing desired manifestations for a computer user.”).) 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 (“A service, accessible on the network, is provided for locating the handle associated with a digital object. The handle is used to obtain a pointer to the network location of...the digital object.”), 7:2-20 (“**A handle server, accessible on the network, provides the pointer in response to the presentation of a handle. Multiple servers provide the service, each serving a portion of the handle space....**Hash values are computed on the handles and the hash values are distributed among **multiple handle servers, each handle server having a table which associates handles**

with pointers.”), 21:6-24 (“[M]ultiple servers...provide...handle-to-pointer translation services (called ‘handle servers’)...It should also be clear that services can be provided on a widely distributed basis by **multiple similar servers at different locations on the network**, or by single centralized servers.”), 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles....”).)

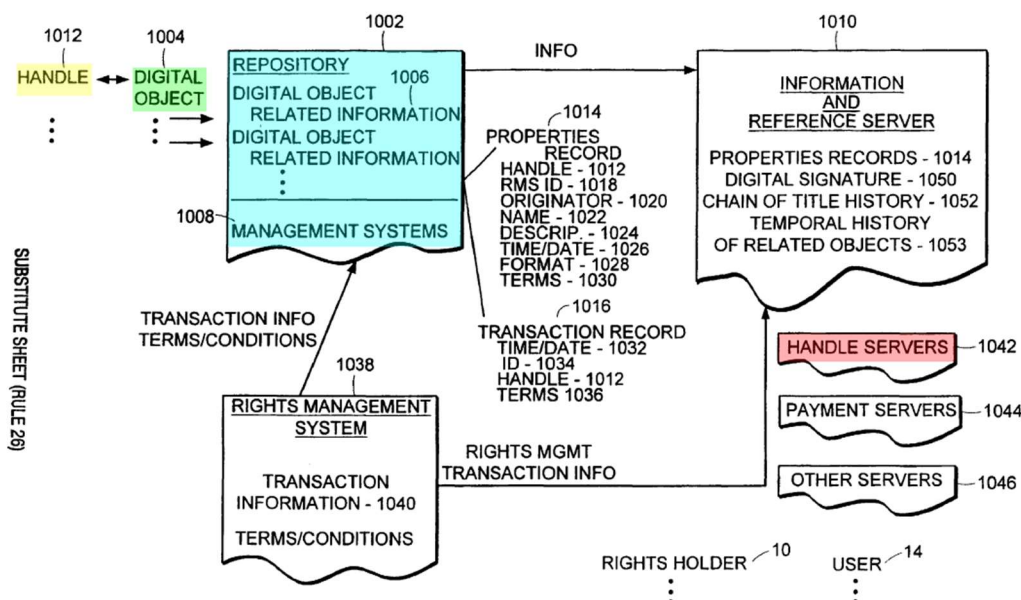


FIG. 1

(*Id.*, FIG. 1.)

54. Kahn discloses that the handle servers (“a plurality of location servers”) provide one or more handles and associated pointers responding to a client’s request to retrieve the 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 (“Given a handle, **the following steps, shown in Figure 8, are followed to obtain a set of pointers associated with the handle.**”), 46:29-47:25; *see also id.*, 72:32-74:32)

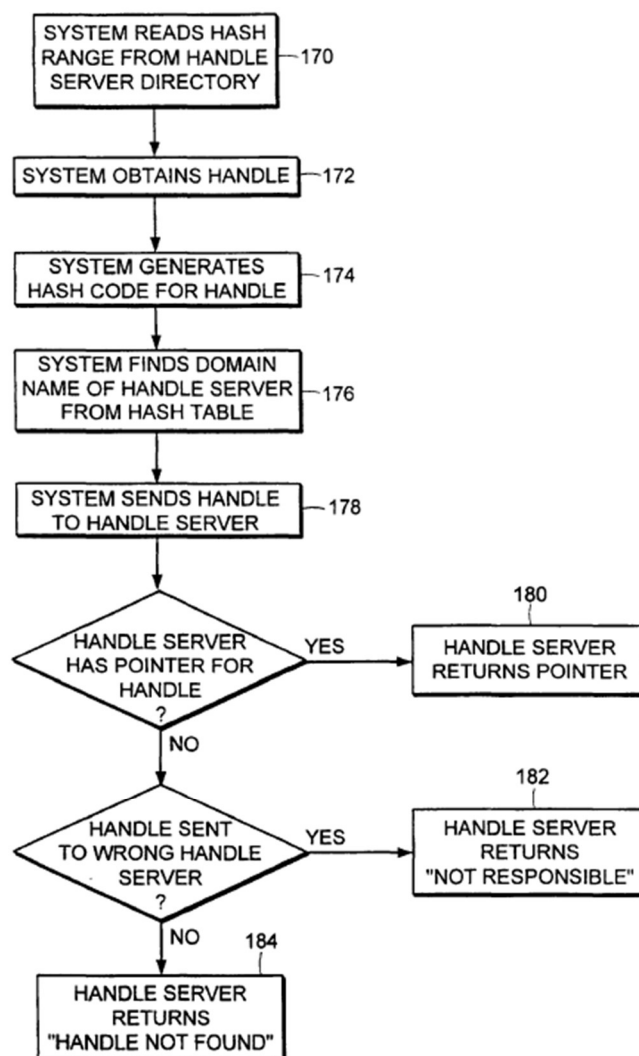


FIG. 8
SUBSTITUTE SHEET (RULE 26)

(*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 (or a hash

code/handle server table, *see* 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 (“The handle server directory...**allows users to find the correct handle server for processing a given handle....**The handle server directory...distributes handles to handle servers based upon a hash of handles. The handle server directory provides a list of domain names of handle servers and identifies the set of hash values of handles which each of the handle servers can map to pointers.”), 38:9-17 (“[T]he handle server system...includes handle servers...and handle directory servers...located at certain well known locations. **The directory servers will maintain a table of network addresses of handle servers (generally, each handle server will contain such a directory).** **This table will generally be downloaded by each participating site frequently enough to be ‘acceptably’ up-to-date at all times.**”), 46:22-47:5 (“In one example, the handle server directory...holds a table...which associates hash ranges...with domain names of handle servers...(Figure 7)....In a first step 170, a client system downloads the table that associates hash ranges with handle server domain names from the handle server directory for future use. The client also can omit this step if it has previously stored the table; frequent changes in the table

may necessitate doing this every time... **The system** then generates the hash code for the handle using a predetermined hashing algorithm (step 174) and **consults the hash range/handle-server-domain-name table to determine the domain name of the appropriate handle server (step 176).**”), 73:4-74:26 (“The system requesting the digital object obtains...the hash code/handle server table from the handle server directory.... This is done during the system’s initialization....Once the handle is obtained, the system that retrieves the object ‘hashes’ the object’s handle and uses this hash value to perform a table lookup in the **hash code/handle server table**....The requesting system sends a request-for-pointer-information UDP packet...to the handle server. One or more pointers, one returned, identifies the network location of the one or more repositories (if one is associated with the object)....For each pointer returned by the handle server, the requesting system uses the pointer to attempt to obtain a copy of the digital object....”).)

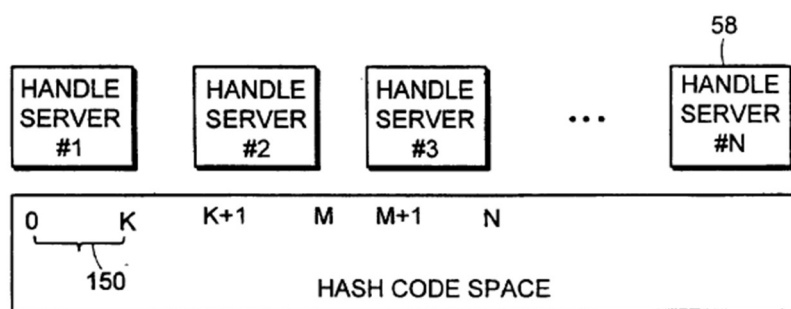


FIG. 7

149 ↗

(*Id.*, FIG. 7.) As shown in Figure 7 above, each handle server is assigned a range of hash codes/values 150. (*Id.*, 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 (“**The system then generates the hash code for the handle using a predetermined hashing algorithm (step 174) and consults the hash range/handle-server-domain-name table to determine the domain name of the appropriate handle server (step 176).** The system subsequently sends the handle to the handle server as part of a request pointer information UDP packet (step 178). The handle server then returns its response to the requesting system. **If the handle server found the pointers, a list of pointers is returned (step 180).**”), 73:17-26 (“Once the handle is obtained, the system that retrieves the object ‘hashes’ the object’s handle and **uses this hash value to perform a table lookup in the hash code/handle server table....The requesting system sends a request-for-pointer-information UDP packet...to the handle server. One or more pointers, once returned, identified the network location of the one or more repositories (if one is associated with the object)....**”).)

55. Returning to Figure 8, the client in step 172 receives a handle from a user's request for obtaining a certain digital object (*Id.*, 46:34-47:1 ("In a later step 172, assume the system obtains a handle for which pointers are desired."), 73:8-16 ("**A user (or more likely, his software agent) obtains a handle...for an object....**The handle may be obtained as part of a result of a bibliographic search or be provided by some other electronic means such as an electronic reference list in another object, or by scanning a barcoded sequence on paper. The system that is retrieving the digital object is referred to as the requesting [client] system....").) 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 ("Once the handle is obtained, the system that retrieves the object 'hashes' the object's handle and uses this hashed value to perform a table lookup in the hash code/handle server table....The requesting system sends a request-for-pointer-information UDP packet...to the handle server.").)

56. 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. (*Id.*, 47:10-11 (“**If the handle server found the pointers**, a list of pointers is returned (step 180).”), 73:32-74:12 (“**The handle server verifies that the handle falls within the set of handles for whose hash value it is responsible....If it is not in this set, due to some dynamic system change or error condition**, the handle server sends an invalid-handle-server-selected response UDP packet to the requestor....If the requesting system receives an invalid-handle-server-selected response UDP packet, it refreshes its hash code/handle server table [shown in Figure 7] from the handle server directory, and the requesting system repeats prior steps. **This will typically be needed only if the table has changed between the time the table was downloaded and the actual request was made.**”).) 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 system in step 180. (*Id.*, 29:8-10 (“A handle server accepts a handle as input and **returns a list of pointers** associated with the handle....”), 47:10-13 (“If the handle server found the pointers, **a list of pointers is returned (step 180)**. This could include pointers to use one or more repositories....”), 73:23-26 (“**One or more pointers, once returned**, identifies the network location of the one or more repositories (if one is associated with the object)....”), 74:21-26 (“An object may be stored in several repositories. Multiple pointers to these repositories may be

returned to the requesting system. For **each pointer returned by the handle server**, the requesting system uses the pointer to attempt to obtain a copy of the digital object....”).) The client then uses the pointer(s) to obtain the digital object stored in a repository. (*Id.*, 74:21-29 (“An object may be stored in several repositories. Multiple pointers to these repositories may be returned to the requesting system. **For each pointer returned by the handle server, the requesting system uses the pointer to attempt to obtain a copy of the digital object....**If a copy is successfully obtained from one repository, the rest of the pointers will generally be ignored.”).)

57. Kahn discloses the claimed “location server[]” under the Interpretation from Other Proceedings, i.e., “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.” (Section VIII; Ex. 1001, 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). (Ex. 1005, 7:5-7 (“**A handle server, accessible on the network**, provides the pointer in response to the presentation of a handle.”), 21:21-24 (“**services can be provided on a widely distributed basis by multiple similar servers at different locations on the network**”), 73:27-31 (“...many handle servers [are]

distributed on a network without a central nodal point in the system (for reliability.”).) As discussed for limitation [1.a], each handle is a claimed “identifier” that uniquely identifies a digital object, under the Interpretation from Other Proceedings of “identifier.” (See below Section IX.A.1.ii; *see also* Ex. 1005, 21:3-5; *see also id.*, 6:27-28 (“**The handle for a digital object is unique** both across the network and over time.”), 9:23-26 (“**The handles used to uniquely identify digital objects** are designed to be extensible and expandable to accommodate virtually any number objects over many years.”), 11:8-13 (“Formally, a digital object is an instance of an abstract data type that has two components, data and key-metadata....The key-metadata includes **a handle, i.e., an identifier globally unique to the digital object....**”).) And as discussed above, Kahn discloses that pointers are locations which identify the location of a digital object. Kahn further discloses that each handle server maintains mappings between handles (i.e., identifiers) and pointers (i.e., locations). (Ex. 1005, 7:17-20 (“...multiple handle servers, each handle server having a table which associates handles with pointers.”), 21:6-10 (“[M]ultiple servers...provide...handle-to-pointer translation services (called ‘handle servers’)....”), 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a

subset of all handles....”).) And as discussed above, Kahn’s pointer discloses a “location” (or similar terms) as described in the ’978 patent. (Ex. 1001, 4:28-30.)

58. 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 person of ordinary skill in the art would have understood that the returned pointers correspond to mappings of the handle to the pointers, consistent with the ’978 patent’s disclosure. (Ex. 1001, 2:32-37, 4:61-64.) Moreover, Kahn discloses that the full handle record (i.e., including the handle itself and all corresponding data) may be returned in response to a request from the client. (Ex. 1005, 42:1-11 (“We now describe how a client...resolves any handle....**[T]he client request may ask for all the data in a handle record....**”), 44:16-21 (“To resolve a handle, a handle server receives as input a handle and returns some or all of the fields of typed data in the corresponding handle record. **The client can request that all data fields in the handle record be returned....**”).) Thus, a person of ordinary

skill in the art would have understood that if pointers are returned to the client system from the handle server, the handle itself may also be returned in response to the location request message from the client. 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). (*Id.*, 43:31-44:4, 45:22-46:24; Ex. 1001, 10:1-4.) Thus, Kahn discloses the claimed “location server[]” under the Interpretation from Other Proceedings.

59. Kahn discloses the claimed “plurality of location servers” under the Interpretation from Other Proceedings for “plurality of location servers,” i.e., “location servers in non-hierarchical structures.” (*See* above Section VIII.) A person of ordinary skill in the art would have understood that the handle servers (each a “location server”) 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. Indeed, Kahn discloses that the hash function is used to distribute handles and associated pointers among the handle servers. (Ex. 1005, 4:19-22, 7:8-9, 37:31-38:3, FIG. 7.) A person of ordinary skill in the art 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.

60. Kahn also discloses the claimed "location information" under the Interpretation from Other Proceedings, i.e., "one or more identifiers and their associated locations." (*See* above Section 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" under the Interpretation from Other Proceedings) in response to a client's request to retrieve the locations of digital objects ("providing location information to location queries").

61. I have been asked to consider a scenario where 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 Interpretation from Other Proceedings for "location information." In that scenario, a person of ordinary skill in the art would have been motivated to include such a feature in Kahn. A person of ordinary skill in the art 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 used 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. (Ex. 1005, 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 the client to the handle server also includes the same handle. (*Id.*, 16:6-8 (“The properties record, the transaction record, and the digital object all are normally accessible using the handle.”), 46:34-47:8 (“In a later step 172, assume the [client] system obtains a handle for which pointers are desired....The system subsequently sends the handle to the handle server as part of a request pointer information UDP packet (step 178).”), 73:8-22 (“A user (or more likely, his software agent) obtains a

handle...for an object....The handle may be obtained as part of a result of a bibliographic search or be provided by some other electronic means such as an electronic reference list in another object, or by scanning a barcoded sequence on paper. The system that is retrieving the digital object is referred to as the requesting [client] system....Once the handle is obtained, the system that retrieves the object ‘hashes’ the object’s handle and uses this hashed value to perform a table lookup in the hash code/handle server table....The requesting system sends a request-for-pointer-information UDP packet...to the handle server.”.) Thus, a person of ordinary skill in the art would have been motivated to include the claimed feature.

62. A person of ordinary skill in the art 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. (*Id.*, 7:17-20 (“...multiple handle servers, each handle server having a table which associates handles with pointers.”), 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles....”).) 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 (“When a digital object is accessed..., the recipient

receives...the key-metadata of the digital object”), 23:33-34 (“the key-metadata (including, of course, the handle) of the digital object is included”) and thus a person of ordinary skill in the art 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. Accordingly, Kahn discloses and/or suggests this limitation.

- ii) **[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”**

63. Kahn discloses this limitation. **First**, Kahn discloses “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.” For example, 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”) (Kahn’s 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 the pointer collectively indicate the location of an object (“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. (Ex. 1005, 7:17-20 (“...multiple handle servers, each handle server having a table which associates handles with pointers.”), 21:6-10 (“[M]ultiple servers...provide...handle-to-pointer translation services (called ‘handle servers’)....”), 28:31-34 (“A handle for an object may be associated with zero or more object pointers. Object pointers contain location information for location digital objects....”), 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles....”).) While Kahn discloses that a handle may be associated with zero pointers when an object is a physical object, a person of ordinary skill in the art 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 (“A handle server accepts a handle as input and returns a list of pointers associated with the handle, where each pointer = {domain name of storage system (repository), domain name of RMS}....The domain name of the storage system may be null if the rights stored in the RMS do not include obtaining a copy of the object, or if the rights apply to a ‘physical’ object.”), 37:23-24 (“The pointer to the repository may be null if the object is not available on-line.”).) 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 the associated pointers indicate the location of the object in the system. (*Id.*, 9:23-26 (“**The handles used to uniquely identify digital objects** are designed to be extensible and expandable to accommodate virtually any number objects over many years.”), 21:3-5 (“Each digital object has a **‘handle’, a concise unique identifier for a digital object** used for storage and retrieval operations and other repository functions.”), 28:31-34 (“A handle for an object may be associated with zero or more object pointers. Object pointers contain location information for locating digital objects....”).)

64. **Second**, Kahn discloses “wherein the identifier identifies an entity and the location string specifies a location of data pertaining to the entity.” For example, 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. (*Id.*, 7:4-5 (“The handles comprise unique character strings associated with the servers which generated them.”).) 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.” (*Id.*, 21:3-5; *see also id.*, 6:27-28 (“**The handle for a digital object is unique** both across the network and over time.”), 9:23-26 (“**The handles used to uniquely identify digital objects** are designed to be extensible and expandable to accommodate virtually any number objects over many years.”), 11:8-13 (“Formally, a digital object is an instance of an abstract data type that has two components, data and key-metadata....The key-metadata includes **a handle, i.e., an identifier globally unique to the digital object....**”).)

65. Kahn further discloses one or more pointers (“location string”) indicating the location of a sequence of bits/digits (“data”) pertaining to an object (“specifies a location of data pertaining to the entity”) in the network. Kahn discloses that an object (“entity”) is represented by a sequence of bits/digits (“data”) stored in different repositories. (*Id.*, 1:10-12 (“By a ‘digital object’ we broadly mean **any set of sequences of bits or digits** and an associated unique identifier which we call a ‘handle.’”), 9:23-26 (“The handles used to uniquely

identify digital objects are designed to be extensible and expandable to accommodate virtually any number of objects over many years.”), 11:16-18 (“A ‘repository’...is a digital storage system into which digital objects...may be placed and retained for possible subsequent retrieval.”).) A person of ordinary skill in the art would have understood that sequences of bits/digits associated with each digital object are the claimed “data.” 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 (“By a ‘digital object’ we broadly mean **any set of sequences of bits or digits** and an associated unique identifier which we call a “handle.”), 4:25-29 (“There are multiple... servers each of which accepts identifiers [i.e., handles] of a subset of the digital objects and returns **corresponding pointers to the locations of the digital objects** in the network.”), 6:30-35 (“**The handle is used to obtain a pointer to the network location...of the digital object.**”), 28:31-32 (“A handle for an object may be associated with zero or more object pointers. **Object pointers contain location information for locating digital objects.**”), 29:8-12 (“A handle server accepts a handle as input and returns a list of pointers associated with the handle, where each **pointer = {domain name of storage system (repository),**

domain name of RMS}.”.) Thus, Kahn discloses “wherein the identifier identifies an entity and the location string specifies a location of data pertaining to the entity.” Accordingly, Kahn discloses this limitation.

66. Kahn’s handle is an “identifier” under the Interpretation from Other Proceedings, i.e., “a unique encoding that identifies an individual entity, and with which zero or more data location specifiers are associated in a location server.” (See above Section 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). A person of ordinary skill in the art 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. (See also Ex. 1005, 33:3-14 (“Handles should be globally unique across the network and over time; should be essentially permanent, since rights on an object may last many years; should not have any location information encoded in the identifier’s namespace, since an object may be located at multiple and changing locations over time....”), 33:22-27 (“The authorization and rules for creating a handle are determined on a country-by-country basis. In one scheme, as seen in Figure 5, handles are printable strings...having a country code...appended to a variable length string defined on a per country basis....”).) 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 interpreted in the Other Proceedings). (*See* Section IX.A.1.i.)

- iii) **[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”**

67. Kahn discloses this limitation. **First**, Kahn discloses “a second location server comprising a second set of location information.” As discussed for limitations [1.pre] and [1.a], Kahn discloses multiple handle servers, where 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. (*See* Sections 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 one or more handles and their associated pointers (“second set of location information”). (*See* Sections IX.A.1.i-ii; Ex. 1005, 4:25-29 (“There are multiple... servers each of which accepts identifiers [i.e., handles] of a subset of the digital objects and returns corresponding pointers to the locations of the digital objects in the network.”), 6:30-35 (“The handle is used to obtain a pointer to the network location...of the

digital object.”), 7:4-20 (“A handle server, accessible on the network, provides the pointer in response to the presentation of a handle. Multiple servers provide the service, each serving a portion of the handle space....Hash values are computed on the handles and the hash values are distributed among multiple handle servers, each handle server having a table which associates handles with pointers.”), 28:31-32 (“A handle for an object may be associated with zero or more object pointers. Object pointers contain location information for location digital objects....”).)

68. **Second**, Kahn discloses “wherein at least a portion of the second set of location information differs from the first set of location information.” For example, Kahn explains that each handle is unique to a digital object. (Ex. 1005, 21:3-5 (“Each digital object has a **‘handle’**, a **concise unique identifier for a digital object** used for storage and retrieval operations and other repository functions.”).) Further, the second handle server stores a handle and pointer for a second digital object. (*Id.*, 4:25-29 (“There are multiple... servers each of which accepts identifiers [i.e., handles] of a subset of the digital objects and returns corresponding pointers to the locations of the digital objects in the network.”), 37:31-38:3 (“Handle servers have the following characteristics: a **handle server holds pointers associated with a subset of all handles**; handles are assigned to handle servers based upon hash values computed on the handles; handle servers are

assigned ranges of hash values; the set of all hash values is partitioned among the set of all handle servers. This leads to a highly efficient and reliable mechanism for locating objects and from handles.”.) 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”). Accordingly, Kahn discloses this limitation.

- iv) **[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.”**

69. Kahn discloses this limitation. **First**, Kahn discloses “programming logic stored on each of the location servers.” For example, 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.” (Ex. 1005, 26:28-31 (“[A]n example system...includes a digital object management system...which includes hardware and **software to create and store digital objects and manage rights to the objects.**”), 42:2-3 (“[E]ach **handle server can be implemented as one or more server computers....**”); *see*

also id., 21:6-10 (“**The overall system...comprises multiple servers which provide...handle-to-pointer translation services (called ‘handle servers’).**”), 29:5-12 (“Handle servers...process handle query requests. If the handle which is the subject of the query is found by a handle server, the object pointers associated with the handle are returned to the requesting client. A handle server accepts a handle as input and returns a list of pointers associated with the handle, where each pointer = {domain name of storage system (repository), domain name of RMS}.”), 44:6-7 (“A handle is sent to a handle server to locate network addresses of repositories containing that object.”).) Kahn further discloses that the handle system interfaces with the handle servers using routines written in the C programming language. (*Id.*, 40:31-41:2 (“The handle system provides a client library of routines...currently written in the C programming language....They can be included in client programs that wish to contact handle servers.”).) As such, a person of ordinary skill in the art would have understood that each handle server (“location server”), which may be implemented using “hardware and software” and as a computer, performs associated processes using certain configured “programming logic” stored on each handle server.

70. **Second**, Kahn discloses “programming logic...responsive to a location query identifying a desired entity to return a location message.” For

example, Kahn discloses that a handle server (having “programming logic” stored and configured thereon) responds to a client’s request (“responsive to a location query”) that identifies a desired object (“identifying a desired entity”) by returning a message containing a list of pointers (“return a location message”) indicating the location of the digital object. Kahn discloses a client’s request (“location query”) for retrieving the location of a digital object by providing a handle that identifies a desired object (“identifying a desired entity”) to an appropriate handle server. (*Id.*, 46:34-47:8 (“In a later step 172, assume the [client] system obtains a handle for which pointers are desired....The system subsequently sends the handle to the handle server as part of a request pointer information UDP packet (step 178).”), 73:8-22 (“A user (or more likely, his software agent) obtains a handle...for an object....The handle may be obtained as part of a result of a bibliographic search or be provided by some other electronic means such as an electronic reference list in another object, or by scanning a barcoded sequence on paper. The system that is retrieving the digital object is referred to as the requesting [client] system....Once the handle is obtained, the system that retrieves the object ‘hashes’ the object’s handle and uses this hashed value to perform a table lookup in the hash code/handle server table....The requesting system sends a request-for-pointer-information UDP packet...to the handle server.”).) In particular, a user may

request a search for a digital object by using an object's handle as input to the client system, (*id.*, 73:8-14 (“A user (or more likely, his software agent) obtains a handle...for an object....The handle may be obtained as part of a result of a **bibliographic search or be provided by some other electronic means such as an electronic reference list in another object, or by scanning a barcoded sequence on paper.**”)), and the client then performs various processes to locate the digital object associated with the object's handle. (*See* above Section IX.A.1.i.) Specifically, the client system “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” that may contain a pointer to the digital object. (Ex. 1005, 47:1-7; *see also id.*, 73:17-26; *see also* Section IX.A.1.i.) Once the correct handle server is determined, a handle query in the form of a request-for-pointer-information UDP packet (“location query”) is sent from the client system to the correct handle server (having “programming logic” stored and configured thereon). (*See* Section IX.A.1.i; Ex. 1005, 29:5-12 (“**Handle servers...process handle query requests. If the handle which is the subject of the query is found by a handle server, the object pointers associated with the handle are returned to the requesting client.** A handle server accepts a handle as input and returns a list

of pointers associated with the handle....”), 47:5-11 (“**The [client] system subsequently sends the handle to the handle server as part of a request pointer information UDP packet (step 178).** The handle server then returns its response to the requesting system. If the handle server found the pointers, a list of pointers is returned.”), 73:8-34 (“A user (or more likely, his software agent) obtains a handle...for an object....The handle may be obtained as part of a result of a bibliographic search or be provided by some other electronic means such as an electronic reference list in another object, or by scanning a barcoded sequence on paper. The system that is retrieving the digital object is referred to as the requesting [client] system....Once the handle is obtained, the system that retrieves the object ‘hashes’ the object’s handle and uses this hashed value to perform a table lookup in the hash code/handle server table....The requesting system sends a request-for-pointer-information UDP packet...to the handle server.... The handle server verifies that the handle falls within the set of handles for whose hash values it is responsible....”); *see also id.*, 7:5-7 (“A handle server, accessible on the network, provides the pointer in response to presentation of a handle.”), 43:20-23 (“[A] user can present a handle to a handle server and be advised of some repository which...contains the digital object designated by the handle.”), 44:6-7 (“A handle is sent to a handle server to locate network addresses of repositories containing that

object.”.) Then, “[i]f the handle server found the pointers, a list of pointers is returned....” (*Id.*, 47:9-11 (“The handle server then returns its response to the requesting system. If the handle server found the pointers, a list of pointers is returned (step 180).”); *see also id.*, 73:21-27 (“The requesting system sends a request-for-pointer-information UDP packet...to the handle server. One or more pointers, once returned, identifies the network location of the one or more repositories....”), FIG. 8 (step 180); Section IX.A.1.i.) Thus, Kahn discloses “programming logic...responsive to a location query identifying a desired entity to return a location message.”

71. **Third**, Kahn discloses “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.” For example, 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 that received the client’s request (“location server receiving the location query”) contains the handle and associated pointers associated with the requested object (“location information for the desired entity”). Kahn explains 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 system. (Ex. 1005, 47:10-11 (“If the handle server found the pointers, a list of pointers is returned....”), 73:21-74:26 (“The requesting system sends a request-for-pointer information UDP packet...to the handle server. One or more pointers, one returned, identifies the network location of the one or more repositories (if one is associated with the object)....For each pointer returned by the handle server, the requesting system uses the pointer to attempt to obtain a copy of the digital object....”).) Accordingly, Kahn discloses this limitation.

2. **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.”**

72. Kahn discloses claim 3. **First**, Kahn discloses “wherein the programming logic further comprises logic responsive to the location query to return one of a location message or a redirect message.” 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"). (*See* Section 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.

73. **Second**, Kahn discloses "wherein the location server receiving the query returns the location message if the queried location server contains location information for the desired entity." For example, 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 one or more pointers for the object ("returns the location message if the queried location server contains information for the desired entity"). In particular, the returned message contains a list of pointers ("location message") indicating the locations of an object. (*See* Section IX.A.1.iv.) This message is sent if the handle server does indeed contain the handle and associated one or more pointers for the object. (Ex. 1005, Abstract ("Other aspects include providing multiple servers...each of which accepts identifiers of a subset of the digital objects and returns corresponding pointers to the locations of the digital objects in the network...."), 29:8-12 ("A handle server accepts a handle as input and returns a list

of pointers associated with the handle....”), 47:9-10 (“If the handle server found the pointers, a list of pointers is returned.”), 73:32-74:26 (“The handle server verifies that the handle falls within the set of handles for whose hash value it is responsible....If the handle server is responsible for the handle, it verifies that the handle is present in its database....Multiple pointers to these repositories may be returned to the requesting system. For each pointer returned by the handle server, the requesting system uses the pointer to attempt to obtain a copy of the digital object....”).)

74. **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.” I have been asked to assume that the claimed “redirect message” is met by a cache miss response due to stale or incorrect location information. Applying this assumption, which I refer to as the “Applied Interpretation,” Kahn discloses this limitation in several ways.

75. For example, Kahn discloses that the queried handle server (“queried location server”) transmits a not-responsible-for-handle message (“returns a redirect message”) to the client system 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”). (Ex. 1005, 47:13-15 (“[I]f the handle was sent to the wrong handle server, it returns a **not-responsible-for-handle message** (step 182).”), FIG. 8 (step 182); *see also id.*, 73:1-74:26 (explaining the same lookup procedure in further detail but referring to the not-responsible-for-handle message by the type of packet, i.e., invalid-handle-server-selected response UDP packet, which is sent if the handle server is not responsible for the handle), 73:32-74:4 (“The handle server verifies that the handle falls within the set of handles for [which] it is responsible 750. **If it is not in this set, ...the handle server sends an invalid-handle-server-selected response UDP packet to the requestor 752.**”), FIG. 25.)

76. 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 (“If the handle

was sent to the wrong handle server, it returns a not-responsible-for-handle message (step 182). In this case, the client system should download the hash range/handle-server-domain-name table [shown in Fig. 7] from the handle server directory again and attempt the mapping again. The requester will determine how many times to try before giving up...."); *see also id.*, 74:5-9 ("If the requesting system receives an invalid-handle-server-selected response UDP packet, it refreshes its hash code/handle server table [shown in Fig. 7] from the handle server directory, and the requesting system repeats prior steps.".) Accordingly, under the Applied 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. (*See also* Section IX.C.1 (addressing disclosure of "redirect message" under other interpretations).) Accordingly, Kahn discloses claim 3.

3. 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."

77. Kahn discloses claim 5. As discussed for limitations [1.pre]-[1.b], Kahn discloses each of the multiple 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. (See Sections IX.A.1.i-iii; Ex. 1005, 4:19-22 (“The identifiers are generated in a manner **to distribute the pointer information with the unique identifier information[] relatively evenly among the servers**, using a hashing algorithm.”), 7:2-20 (“**A handle server, accessible on the network**, provides the pointer in response to the presentation of a handle. **Multiple servers provide the service, each serving a portion of the handle space....**Hash values are computed on the handles and the hash values are distributed among **multiple handle servers**, each handle server having a table which associates handles with pointers.”), 37:31-38:3 (“Handle servers have the following characteristics: **a handle server holds pointers associated with a subset of all handles**; handles are assigned to handle servers based upon hash values computed on the handles; handle servers are assigned ranges of hash values; **the set of all hash values is partitioned among the set of all handle servers.** This leads to a highly efficient and reliable mechanism for locating objects and from handles.”).)

78. As a person of ordinary skill in the art would have understood, in order to maintain information, including the handles and their associated pointers (collectively, “location information”) for locating digital objects, on each of the

handle servers (“location servers”), such information would have necessarily been stored on a persistent storage device. Otherwise, such information for locating/tracking the digital objects would have been lost after the handle server is powered off. Indeed, Kahn discloses that the handle servers store “items that must be retained over very long periods of time” (Ex. 1005, 38:18-28) and the servers may “be used to back up each other” (*id.*, 39:7-9). Thus, Kahn discloses claim 5.

4. 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.”

79. Kahn discloses claim 6. **First**, Kahn discloses “the location information in the location server is maintained in an indexed location store.” For example, Kahn discloses that one or more handles and associated pointers (collectively, “location information”) in a handle server (“location server”) are maintained in a table in the handle server associating the handles with pointers (“indexed location store”). In particular, Kahn explains that each handle server has a table that associates handles with pointers. (Ex. 1005, 7:17-20 (“Hash values are computed on the handles and the hash values are distributed among multiple handle servers, each handle server having a table which associates handles with pointers.”), 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles....”).) A person of

ordinary skill in the art 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 one or more pointers and associated handles (“location store”). (Section IX.A.3.) A person of ordinary skill in the art would have further understood that the table in the handle server is an “indexed location store” because each handle may be used as an index to retrieve the associated one or more pointers (part of the “location information”).

80. **Second**, Kahn discloses “an indexed location store indexed by a hash table.” For example, as discussed above, 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”). (Ex. 1005, 47:1-11 (“**The system** then generates the hash code for the handle using a predetermined hashing algorithm (step 174) and **consults the hash range/handle-server-domain-name table to determine the domain name of the appropriate handle server (step 176). The system subsequently sends the handle to the handle server** as part of a request pointer information UDP packet (step 178). The handle server then returns its response to the requesting system. **If the handle server found the pointers, a list**

of pointers is returned (step 180).”), 73:17-27 (“Once the handle is obtained, the system that retrieves the object ‘hashes’ the object’s handle and **uses this hash value to perform a table lookup in the hash code/handle server table....The requesting system sends a request-for-pointer-information UDP packet...to the handle server. One or more pointers, once returned, identified the network location of the one or more repositories (if one is associated with the object)....”).)**

Specifically, the client system “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. (*Id.*, 47:1-5.) Put differently, a client system applies a hash algorithm to the handle resulting in a hash code/value used to index into the hash range/handle-server-domain-name table in order to determine/calculate the location of a handle server and its table storing sets of handles and associated pointers. (*Id.*) Accordingly, Kahn discloses claim 6.

5. Claim 10

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

81. I understand that this limitation is a preamble of the claim and I have been asked to assume that the preamble of claim 10 is limiting. Under such an assumption, Kahn discloses and/or suggests this limitation for the same reasons as discussed for limitation [1.pre]. (*See* Section IX.A.1.i.)

- ii) **[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”**

82. Kahn discloses this limitation. **First**, Kahn discloses “a plurality of location servers containing location information corresponding to a plurality of entities.” 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”). (*See* Sections IX.A.1.i-ii.)

83. **Second**, Kahn discloses “the location information formatted according to a transfer protocol configured for manipulating 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.” (Ex. 1005, 26:8-14, FIG. 33 (showing a diagram of portions of a handle system).) Thus, a person of ordinary skill in the art 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 system) (“a transfer protocol configured for manipulating location information”).

84. **Third**, Kahn discloses “the location information...comprising at least one application server address.” Kahn discloses multiple handle servers each storing sets of handles and one or more associated pointers (collectively, “location information”). (*See* Sections IX.A.1.i-ii.) A pointer contains the domain name of the repository which contains the associated digital object. (Ex. 1005, 29:5-12 (“Handle servers...process handle query requests. If the handle which is the subject of the query is found by a handle server, the object pointers associated with the handle are returned to the requesting client. A handle server accepts a handle as input and returns a list of pointers associated with the handle, where **each pointer**

= {**domain name of storage system (repository), domain name of RMS**}.”.)

Additionally, Kahn discloses that there are multiple repositories, each repository containing digital objects for users to access through a client’s request (i.e., location query) to retrieve the location of a digital object. (*Id.*, 6:22-24 (“Owners of rights in digital objects may deposit copies of the digital objects in the repository, via the network. There may be multiple repositories.”), *see* Sections IX.A.1.i, IX.A.1.iv.) Kahn discloses that each of these repositories 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. (*Id.*, 11:22-27 (“The repository has mechanisms for adding new digital objects to its collection (depositing) and for making them available (accessing), using, at a minimum, a so-called repository access protocol. The repository may contain other related **information, services and management systems**.”), 13:2-4 (“**[A] repository may offer a service** in which it deposits a digital object and registers the handle simultaneously...”), 22:11-23 (“A ‘simple repository access protocol’ (RAP) is supported by each repository and discussed below. The RAP may be merely a subset of a larger interface protocol used by repositories, provided that the functions or operation of the RAP not be affected by any implemented supersets of the protocol. In particular,...the RAP...allows for

accessing a stored digital object or its metadata by specifying its handle, a service request type, and additional parameters. If this request is complied with, the output of the service request is termed a ‘dissemination’.... A dissemination is the result of an access service request, along with additional data affixed to it.”.) A person of ordinary skill in the art would have understood that each of the multiple repositories would have implemented a server (i.e., application server) in order to provide the programming, services, information storage, and management systems required for implementing a repository access protocol. Thus, a person of ordinary skill in the art would have understood that the one or more pointers (part of the “location information”) provide an “application server address” as claimed because a pointer indicates the domain name of a repository, and a repository has a server with an address. Thus, Kahn discloses “the location information...comprising at least one application server address.”

85. **Fourth**, Kahn discloses “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.” As discussed for limitation [1.pre], a person of ordinary skill in the art would have understood that the handle servers (“the plurality of location servers”) are arranged in a cluster structure (“arranged in a cluster topology”), rather than a hierarchical

or tree-like structure. (*See* Section 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. (Ex. 1005, 37:31-38:3 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles; handles are assigned to handle servers based upon hash values computed on the handles; handle servers are assigned ranges of hash values; **the set of all hash values is partitioned among the set of all handle servers.** This leads to a highly efficient and reliable mechanism for locating objects and from handles.”); *see also id.*, 29:23-26 (“The handle server directory provides a list of domain names of handle servers and **identifies the set of hash values of handles which each of the handle servers can map to pointers.**”).) 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.

- iii) **[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.”**

86. Kahn discloses this limitation. **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. (*See* Section IX.A.1.iv.) As such, a person of ordinary skill in the art would have understood that operations/services provided on the computer are operated using associated “programming logic” stored and configured on each handle server in order to perform each of the operations of the handle server, including those recited in limitation [10.b] and discussed below.

87. **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 system for pointers associated with the handle in the request (“for a desired identifier”). (See Section 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”). (See Section IX.A.2.)

88. **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 system for pointers associated with the handle in the request (“for a desired identifier”). (See Section 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”). (See Section IX.A.2.) Accordingly, Kahn discloses this limitation.

6. Claim 12: “The system of claim 10, wherein the location information comprises network address information.”

89. Kahn discloses claim 12. As discussed for limitation [1.pre], Kahn discloses a handle server maintains sets of handles and associated pointers (“location information”). (See Section IX.A.1.i.) Further, pointers include the domain name (“network address information”) of a repository storing digital objects in a network. (Ex. 1005, Abstract (“The [digital] objects are stored at [repositories] accessible in the network...”), 6:8-35 (“Copies of the digital objects are stored in a repository....The objects are accessed at the repository via the network....The handle is used to obtain a pointer to the **network location of...the**

digital object.”), 11:16-18 (“A ‘repository’ 1002 is a digital storage system into which digital objects 1004 may be placed and retained for possible subsequent retrieval.”), 21:21-24 (“...services can be **provided on a widely distributed basis** by multiple similar servers at different locations on the network...”), 29:5-12 (“Handle servers...process handle query requests. If the handle which is the subject of the query is found by a handle server, the object pointers associated with the handle are returned to the requesting client. A handle server accepts a handle as input and returns a list of pointers associated with the handle, where each **pointer** = {**domain name of storage system (repository)**, domain name of RMS}.”), 73:23-31 (“**One or more pointers, once returned, identifies the network location of the one or more repositories** (if one is associated with the object)...This strategy assures a random distribution of handle server requests among many handle servers distributed on **a network without a central nodal point in the system** (for reliability).”).) 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. (*Id.*)

7. 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.”

90. 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”). (*See* Section IX.A.6.) As discussed for limitation [1.a], the digital object is comprised of sequences of bits/digits (“data”). (*See* Section 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.

8. Claim 14

- i) [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:”**

91. I understand that this limitation is a preamble of the claim and I have been asked to assume that the preamble of claim 14 is limiting. Under such an

assumption, Kahn discloses the limitations therein. **First**, Kahn discloses “[a] method of handling location queries in a network, the network comprising a plurality of location servers” for reasons similar to as discussed for limitation [1.pre]. For example, Kahn discloses a method for handling a client’s requests (“location queries”) to retrieve a digital object in a network (“network”), where the network contains multiple handle servers (“a plurality of location servers”). (*See* Section IX.A.1.i.)

92. **Second**, Kahn discloses “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.” 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]. (*See* Section 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”). (*See* Section IX.A.1.i.)

ii) [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;”

93. 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 system (“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. (*See* Section IX.A.1.iv.) Accordingly, Kahn discloses this limitation.

94. Kahn discloses the claimed “client” under the Interpretation from Other Proceedings, i.e., “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.” (*See* above Section VIII; *see also* Ex. 1001, 4:30-34.) For example, Kahn discloses a user entering a handle into a client system (“client”) to obtain a pointer indicating the location of the digital object in the distributed network. (Ex. 1005, 46:25-47:25, 73:1-74:32.) Moreover, Kahn discloses that the user interface may be “in the form of software running on a workstation.” (*Id.*, 26:28-34 (“[A]n example system...includes a digital object management system...which includes hardware and software to create and store digital objects.... In the system, a user agent (UA)...provides a user interface for

interactions with other elements of the system. **The UA is in the form of software running on a workstation.**”), FIG. 2.) Kahn also discloses that the user interface 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 person of ordinary skill in the art would have recognized that the client system (“client”) is either a hardware and/or software component attached to a network. As further discussed for limitation [1.c], the client system also sends a request (i.e., location request message) to a handle server (i.e., location server) in order to initiate lookup of mappings of handles (i.e., identifiers) to one or more pointers (which identify a digital object’s location). (*See* Section IX.A.1.iv.)

- iii) **[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”**

95. Kahn discloses and/or suggests this limitation. **First**, Kahn discloses “sending a location response message to the client if the queried location server contains location information relevant to an entity identified in the query.” As discussed for limitation [1.c], Kahn discloses sending a message containing a list of pointers (“sending a location response message”) to the client system 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. (*See* Section IX.A.1.iv.)

96. **Second**, Kahn discloses and/or suggests “the location response message comprising location information identifying at least one application server containing information relevant to the entity identified in the query.” 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”). (*See* Section IX.A.1.iv.) Consistent with 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. (*See* Section IX.A.1.i.) As discussed for limitation [10.b], each pointer identifies a repository containing a digital object, and a person of ordinary skill in the art would have understood that Kahn’s repository is an “application server” as claimed. (*See* Section 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”).

- iv) **[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.”**

97. Kahn discloses this limitation. As discussed for claim 3, Kahn discloses that a queried handle server sends a not-responsible-for-handle message (“redirect message”) to the client system (“client”) if the queried handle server (“queried location server”) does not contain handles and associated pointers (collectively “data location information”)⁵ relevant to the object identified in the request (“relevant to the entity identified in the query”). (*See* Section 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

⁵ Kahn discloses “data location information” under the Interpretation from Other Proceedings of “location information” for the same reasons as discussed for limitation [1.pre]. (Sections VIII, IX.A.1.ii.)

correct handle server which the client may use to retrieve the object identified in the query (“redirect message comprising information for finding a location server storing the entity identified in the query”). (*Id.*)

98. I have been asked to consider a scenario where limitation [14.c] does not require that the redirect message be sent from the queried location server. In that scenario, 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. (Ex. 1005, 47:13-19, 74:5-9.) A person of ordinary skill in the art 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” (Ex. 1005, 26:8), and thus a person of ordinary skill in the art would have recognized that information sent between servers in the handle system, e.g., between the handle server directory and the client, as is the case with the transmission of the handle-server-domain-name table, would have been sent as UDP messages. (*See also* Ex. 1014, 1 (“[User Datagram Protocol] provides a procedure for application programs **to send messages** to other

programs with a minimum of protocol mechanism.”); Ex. 1016, 4:37-5:12 (incorporating by reference several internet protocols, including RFC 768 (User Datagram Protocol) [Ex. 1014]).) 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 system to determine the location of the correct handle server that contains a pointer for retrieving the digital object associated with the handle. (See Sections IX.A.1.i, IX.A.2.) Accordingly, Kahn discloses this limitation.

9. 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.”

99. Kahn discloses claim 15. 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. (*See* Section IX.A.1.iv.) As discussed for limitation [1.c], the client system 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.*) Accordingly, Kahn discloses claim 15.

10. 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.”

100. Kahn discloses claim 16. 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.” (Ex. 1005, 7:17-20 (“Hash values are computed on the handles and the hash values are distributed among multiple handle servers, each handle server having a table which associates handles with pointers.”), 21:8-10 (“[M]ultiple

servers...provide...handle-to-pointer translation services (called ‘handle servers’)...”), 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles....”).) A person of ordinary skill in the art 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”).

101. 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. (*See* Section IX.A.9.) Accordingly, Kahn discloses claim 16.

B. Kahn in View of Krasner Discloses and/or Suggests Claims 2 and 11

- 1. 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.”**

102. Kahn in view of Krasner discloses and/or suggests claim 2. Kahn discloses that handle servers store handles and associated one or more pointers (“at least one location”) (*see* Section IX.A.1.i), but Kahn 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 person of ordinary skill in the art would have been motivated to implement this feature in view of Krasner.

103. Like Kahn, Krasner relates to a system of storing data in a client-server architecture. For example, Krasner discloses a system that has location servers that store location information pertaining to a mobile GPS unit. (Ex. 1009, Abstract (“A method and an apparatus for operating a satellite positioning system (SPS) receiver in a client/server architecture.”), 4:10-15 (“The system...includes a location server 25 which communicates through a wireless communication system with the communication system which is coupled to or part of the mobile GPS unit 12. The base station 25 typically includes storage 26 for storing a time sequence of

differential GPS and satellite ephemeris information.”.) Thus, Krasner is similar to Kahn, which discloses a system containing a network of handle servers that store location information pertaining to various digital objects. (See Section IX.A.1.i; *see also* Ex. 1005, Abstract (“A method of managing digital objects (1004) in a network is presented. The objects are stored at [repository] 1002.... Other aspects include providing multiple servers (1042) each of which accepts identifiers of a subset of digital objects and returns corresponding pointers to the locations of the digital objects in the network.”), 7:5-20 (“**A handle server, accessible on the network**, provides the pointer in response to the presentation of a handle. **Multiple servers provide the service, each serving a portion of the handle space**....Hash values are computed on the handles and the hash values are distributed among **multiple handle servers**, each handle server having a table which associates handles with pointers.”), 20:31-21:2 (“The medium of the network (e.g., the cables, airwaves, public switched telephone network) is not shown in the figure; but the network medium could be organized as a single local area network, a wide area network, or a broader network structure (e.g., the Internet).”).) A person of ordinary skill in the art would have therefore been motivated to consider the teachings of Krasner when implementing Kahn’s system/method.

104. Krasner additionally discloses a “system for tracking the location of a mobile GPS receiver over time as it moves.” (Ex. 1009, 3:56-57; *see also id.*, 1:48-51 (“Recently, GPS receivers have been used with radio transmitters, such as a cellular telephone or a mobile telephone in a car to transmit the position of the receiver as it moves.”), 2:39-41 (“[I]t is desirable to provide an improved method and system for providing multiple sets of position information over a period of time through a mobile GPS unit.”).) The mobile GPS unit receives GPS signals at various positions and determines pseudoranges from those signals, which it stores along with a corresponding time stamp in its memory. (*Id.*, 5:49-53.) Krasner’s system includes a location server (or a basestation server) in communication with the mobile GPS unit that is being tracked. (*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., a predetermined period of time has lapsed since the last transmission or an emergency. (*Id.*, 4:66-5:24 (“When the predetermined type event does occur,...the stored pseudoranges and the corresponding time stamps are transmitted via a wireless communication system...to the location server.”); 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.) Thus, Krasner discloses a wireless

communication system in which a location server stores geographical location information (“the at least one location comprises a plurality of geographic locations”), where the geographical location information includes pseudoranges and corresponding time stamps of a mobile GPS unit at various times (“each of the geographic locations representing a position of the entity at a different time”) that may be provided to a user.

105. A person of ordinary skill in the art would have been motivated to include features similar to those disclosed by Krasner when implementing Kahn’s system. Kahn discloses a system in which a pointer (“location”) indicates the location of a digital object in the network. (Ex. 1005, 7:5-20, 21:6-24, 28:32-34, 37:31-33.) Further, Kahn discloses that when one or more pointers associated with a specific handle are retrieved from a handle server, the pointer may be directly transmitted to the client system instead of, for example, using the pointer to retrieve and send the digital object to the client system. (*Id.*, 47:8-13, 74:22-26.) As discussed above, Krasner discloses a system in which location servers (similar to Kahn’s handle servers) store geographical location information and provide that information directly to users. A person of ordinary skill in the art would have recognized that Krasner’s system would beneficially allow a user to track the location of an object over time, while offering advantages over similar systems, as

discussed above. As such, a person of ordinary skill in the art 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.

106. A person of ordinary skill in the art would have also had a reasonable expectation of success combining the teachings of Kahn and Krasner. As discussed above, both references are in the same field of art. (*See above.*) Additionally, implementing the above-discussed features similar to those of Krasner would have involved modifying the Kahn system in a manner that Kahn suggests is feasible—implementing the system such that it stores location information on handle servers, and providing the same according to a request. Moreover, such a combination 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.

2. Claim 11: “The system of claim 10, wherein the location information comprises geographic location information.”

107. The Kahn-Krasner combination discloses and/or suggests claim 11. As discussed for claim 2, the Kahn-Krasner combination discloses and/or suggests

that a handle server (i.e., location server) stores geographical location information (“the location information comprises geographic location”) of a mobile GPS unit that may be provided to a client system. (*See* Section IX.B.1.)

C. Kahn in View of Vingralek Discloses and/or Suggests Claims 3-4, 7, 10, 12-17, 23, and 30

- 1. 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.”**

108. Kahn in view of Vingralek discloses and/or suggests claim 3. As discussed in Section IX.A, Kahn discloses claim 3. (*See* Section IX.A.2.) I have been asked to consider a scenario where Kahn’s redirect message does not disclose “the redirect message comprising information for finding a location server known to have location information relevant to the location query.” In that scenario, a person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek.

109. 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” (Ex. 1007, 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, such that the file grows or shrinks dynamically (*id.*, 9 (“Consider a file consisting of keyed records that is spread across a number of servers connected by a network...A server keeps records of the file primarily on disk and we assume that servers are dedicated to managing a distributed file. We assume that there is a **distributed group of clients that issue requests to insert, delete, and retrieve records for a given key *K* in the file. Consequently, the file grows or shrinks dynamically.**”)). 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.) This approach is closely-related to a distributed version of linear hashing, such as LH*, which “assigns hash buckets to servers and adds servers dynamically upon bucket splits.” (*Id.*, 5, 8.) In both LH* and Vingralek’s approach, clients locate servers for insertion/update/deletion, and perform “key-based retrieval through a hash function whose target domain is dynamically expanded as buckets are split and servers are added, based on the principle of linear hashing.” (*Id.*, 5.) Further, Vingralek discloses that its hash-based distributed file system is implemented

using an address table, which a person of ordinary skill in the art would have understood is akin to a hash table, to look up the server that corresponds to a hashed key. (*Id.*, 4 (“Client requests are serviced by mapping keys onto buckets and looking up the corresponding server in an address table.”), 6 (“[W]e assume an indirection between bucket numbers and server numbers, so that multiple buckets can be assigned to the same server. This indirection is implemented by an *address table*, similar to the directory of the extendible hashing method...but stored in a more compact way.”).) Thus, Vingralek is similar to Kahn, which discloses a hash-based object distribution system and using a handle to locate/retrieve data. (*See* Section IX.A.1.i.). A person of ordinary skill in the art would have therefore been motivated to consider the teachings of Vingralek when implementing Kahn’s system/method.

110. Vingralek additionally discloses propagating updates to clients’ and other servers’ address tables in a deferred manner. (Ex. 1007, 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 (“To limit the communication overhead, clients learn about bucket splits and changes to the server-locating hash function only when a

client request is directed to the ‘wrong’ server,” in which case, “the server forwards the client’s request to the proper server, and notifies the client in order to update the client’s stale information on the hash function.”), 6 (“Similarly to [the] LH* [approach], updates to the address table are propagated to clients and other servers in a deferred manner; forwarding is used to locate buckets when a client has a stale address table.”).) Thus, Vingralek discloses updates to a client’s stale address table are deferred until a wrong first server is reached and the location of a second server needs to be calculated.

111. 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. (*Id.*, 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 (“The client’s and the server’s perception of the file may or may not coincide with the current picture of the file (i.e., they may have stale information on...the address table).”)).) 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.*; *see also id.*, 25 (referring to "client requests and server responses" as "messages").)

112. The key is eventually received by the correct server and the request from the client is executed. (*Id.*, 11.) Via the address table update, "clients gradually bring their perception of the file up-to-date, so that the probability of wrongly addressed messages is significantly reduced." (*Id.*) Vingralek explains that if a server undergoes changes (e.g., if information is moved to another server), the server receives a new address table. (*Id.*, 12-13.) Consistent with the above, a person of ordinary skill in the art would have understood that a server would have up-to-date information at least with respect to its own keys/information, such that the process of updating the client's address table each time a request is forwarded to the next server (until the request reaches a correct server containing the requested information, i.e., "a location server known to have location information relevant to the location query") would have gradually brought the client's

perception of the address table up-to-date. 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.

113. Additionally, Vingralek discloses that “[t]he most salient feature of [the LH*] approach,” where a client’s address table is updated in a deferred manner, “is that its communication overhead is largely independent of the number of servers and clients in the system; thus, it is considered a scalable approach.” (*Id.*, 5.) Thus, Vingralek discloses implementing a redirection mechanism similar to that of the LH* approach and additionally includes improved features to minimize the number of servers and reduce the cost of scaling up a distributed system. (*Id.*, 6, 8.)

114. A person of ordinary skill in the art 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. (Ex. 1005, 38:9-17 (“[T]he handle server system...includes handle servers...and handle directory servers...located at certain well known locations. The directory servers will

maintain a table of network addresses of handle servers (generally, **each handle server will contain such a directory**). **This table will generally be downloaded by each participating site** frequently enough to be ‘acceptably’ up-to-date at all times.”), 46:29-32 (“In a first step 170, **a client system downloads the table that associates hash ranges with handle server domain names from the handle server directory** for future use.”).) 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 (“[A]dditional instances of the digital object at other repositories, and for making changes or withdrawals of such additional instances of the digital objects”), 33:3-14 (“an object may be located at multiple and changing locations over time”), 45:10-12 (“[I]f the location of the repository were to change, the handle server would have to be notified so it could make the corresponding changes....”).) Accordingly, Kahn’s hash table is also subject to change. (*See also id.*, 46:22-34 (“In one example, the handle server directory 59 holds a table 149 which associates hash ranges 150 with domain names of handle servers 58 (Figure 7)....In a first step 170, a client system downloads the table that associates hash ranges with handle server domain names from the handle server directory for future use. The client also can omit this step if it has previously stored the table; **frequent changes in the table** may necessitate doing this every time.”), 74:5-9

(“If the requesting system receives an invalid-handle-server-selected response UDP packet, it **refreshes its hash code/handle server table** from the handle server directory.”).) 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. 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. (*Id.*, 9:23-27 (“**The handles used to uniquely identify digital objects are designed to be extensible and expandable to accommodate virtually any number of objects over many years.** The hashing mechanism provides an efficient and reliable implementation.”), 33:3-14 (“[T]he identifier’s namespace must be variable and unrestricted, **since the number of digital objects created may be expected to increase....**”); Ex. 1007, 4 (“The most salient feature of this approach is that its communication overhead is largely independent of the number of servers and clients in the system; thus, it is considered a scalable approach.”).) Thus, a person of ordinary skill in the art would have been motivated to include deferred hash table updates from the handle server, similar to that of Vingralek, when implementing Kahn’s system/process, so as to reduce the communication overhead

associated with Kahn's existing method of updating the client system's hash table (i.e., frequent downloads from the handle directory server). (Ex. 1005, 46:22-34 ("In one example, the handle server directory 59 holds a table 149 which associates hash ranges 150 with domain names of handle servers 58 (Figure 7)...In a first step 170, **a client system downloads the table that associates hash ranges with handle server domain names from the handle server directory for future use.** The client also can omit this step if it has previously stored the table; **frequent changes in the table may necessitate doing this every time.**".))

115. A person of ordinary skill in the art would have understood that such a modification would have further reduced communication overhead because as discussed for claim 3 in Section IX.A.2, Kahn, similar to Vingralek, also uses a deferred update process via message sent from handle server to client, where the transmission of a not-responsible-for-handle message from a handle server results in the client's download of an updated hash range/handle-server-domain-name table for determining/calculating the location of a correct handle server after reaching a wrong handle server. (*See* above Section IX.A.2.) Combining the transmission of the message from the handle server with the update to the client's hash table would have reduced the number of required interactions between system components, thereby reducing communication overhead.

116. Additionally, a person of ordinary skill in the art 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 system. As disclosed in Kahn, the client system 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. (Ex. 1005, 73:32-34.) Thus, a person of ordinary skill in the art would have been motivated to use a verification method similar to that of Vingralek, i.e., applying the hash function to the server's own hash table, as discussed above, especially because Kahn already discloses that the handle server stores a copy of the hash table. (*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 system,

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 system 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, as discussed in Section IX.A, claim 3. (*See* Section IX.A.2.) A person of ordinary skill in the art 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. Thus, a person of ordinary skill in the art would have been motivated to implement the process of transmitting a redirect message with information for the client to calculate the location of a second handle server. A person of ordinary skill in the art 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 system).

117. A person of ordinary skill in the art would have also had a reasonable expectation of success combining the teachings of Kahn and Vingralek. Both references are in the same field of art and both relate to hash-based distributed computing/file systems. (*See above.*) Moreover, as discussed above, Kahn already utilizes a message to provide updated information to the client system when a wrong server is reached, but would benefit from reducing communication overhead by implementing Vingralek's teaching. A person of ordinary skill in the art would have also had the skill to implement and reasonable expectation of success in achieving such a modification, because it would have involved a combination 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. Accordingly, Kahn in view of Vingralek discloses and/or suggests this limitation.

2. **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.”**

118. The Kahn-Vingralek combination discloses and/or suggests claim 4. 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.” (*See* Section 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.*)

119. 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. (Ex. 1005, FIG. 7, 38:9-17 (“[T]he handle server system...includes handle servers...and handle directory servers...located at certain well known locations. **The directory servers will maintain a table of network addresses of handle servers (generally, each handle server will contain such a directory). This**

table will generally be downloaded by each participating site frequently enough to be ‘acceptably’ up-to-date at all times.”, 46:22-24 (“[T]he **handle server directory 59** holds a **table 149** which associates **hash ranges 150** with **domain names of handle servers 58 (Figure 7).**”), 47:1-21 (“The system then generates the hash code for the handle using a predetermined hashing algorithm (step 174) and consults the hash range/handle-server-domain-name table to determine the domain name of the appropriate handle server (step 176). The system subsequently sends the handle to the handle server as part of a request pointer information UDP packet (step 178). The handle server then returns its response to the requesting system. If the handle server found the pointers, a list of pointers is returned (step 180)...If the handle was sent to the wrong handle server, it returns a not-responsible-for-handle message (step 182). In this case, the client system should download the hash range/handle-server-domain-name table [shown in Fig. 7] from the handle server directory again and attempt the mapping again. The requester will determine how many times to try before giving up....”), 73:17-27 (“Once the handle is obtained, the system that retrieves the object ‘hashes’ the object’s handle and **uses** this hash value to perform a table lookup in the hash code/handle server table.... The requesting system sends a request-for-pointer-information UDP packet...to the handle server. One or more pointers, once

returned, identified the network location of the one or more repositories (if one is associated with the object)...”).)

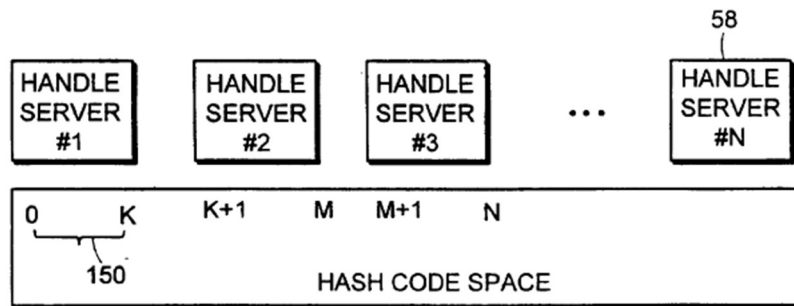


FIG. 7

(*Id.*, FIG. 7.) Consistent with as shown in Figure 7, a person of ordinary skill in the art 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”). (*Id.*)

120. Furthermore, Kahn explains that each handle server domain name in the hash range/handle-server-domain-name table corresponds to a hash range. (*Id.*, 46:22-24 (“[T]he handle server directory 59 holds a table 149 which associates hash ranges 150 with domain names of handle servers 58 (Figure 7).”), 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 person of ordinary skill in the art 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”). Accordingly, Kahn discloses claim 4.

3. **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.”**

121. The Kahn-Vingralek combination discloses and/or suggests claim 7. **First**, Kahn discloses “wherein the location query identifying the desired entity comprises a unique identifier for the desired entity.” As discussed for limitation [1.c], Kahn discloses a client’s request (“location query”) to retrieve the location of a digital object. (*See* Section IX.A.1.iv.) This request, as discussed for limitation [1.pre], includes a client system sending a handle identifying the requested object (“location query identifying the desired entity”) to a handle server. (*See* Section IX.A.1.i; Ex. 1005, 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”). (*See* Section IX.A.1.i; Ex. 1005, 1:16-20, 21:3-5.)

122. **Second**, the Kahn-Vingralek combination discloses and/or suggests “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.” 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. (*See* Section IX.A.1.iv.) As discussed in Section IX.A, claim 6, and in Section IX.C, 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. (*See* Sections IX.A.4, IX.C.1.) In particular, as discussed in Section IX.C, claim 3, the handle server applies the hash function in order to determine if it has the requested pointers associated with the request. (*See* Section IX.C.1.) A person of

ordinary skill in the art 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.” Accordingly, the Kahn-Vingralek combination discloses and/or suggests claim 7.

4. **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.”**

123. The Kahn-Vingralek combination discloses and/or suggests claim 10.

As discussed in Section IX.A, Kahn discloses and/or suggests claim 10. (*See* Section IX.A.5.i-iii.) I have been asked to consider a scenario where 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.” In that scenario, a person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek for reasons similar to those as discussed for claim 3 in Section IX.C. (Section IX.C.1.) For example, as discussed for claim 3 in Section IX.C, 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 in order to determine whether it has the requested pointers associated with the handle. (*See* Section 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 system (“responsive to a location query for a desired identifier to return...a redirect message”) in order 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”). (*See id.*) Accordingly, the Kahn-Vingralek combination discloses and/or suggests claim 10.

5. Claim 12: “The system of claim 10, wherein the location information comprises network address information.”

124. Kahn discloses claim 12 for the same reasons as discussed in Section IX.A, claim 12. (*See* Section IX.A.6.)

6. 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.”

125. Kahn discloses claim 13 for the same reasons as discussed in Section IX.A, claim 13. (*See* Section IX.A.7.)

7. 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.”

126. The Kahn-Vingralek combination discloses and/or suggests claim 14. As discussed in Section IX.A, Kahn discloses and/or suggests claim 14. (*See* Sections IX.A.8.i-iv.) I have been asked to consider a scenario where 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.” In that scenario, a person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek for reasons similar to those as discussed for claim 3 in Section IX.C. (Section IX.C.1.) For example, as discussed for claim 3 in Section IX.C, 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 in order to determine whether it has the requested pointers associated with the handle. (*See* Section 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 system (“sending a redirect message to the client”) in order 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”). (*See id.*) Accordingly, the Kahn-Vingralek combination discloses and/or suggests claim 14.

8. 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.”

127. The Kahn-Vingralek combination discloses and/or suggests claim 15. As discussed in Section IX.A, Kahn discloses claim 15. (*See* Section IX.A.9.) I have been asked to consider a scenario where claim 15 is found to require that the location server is the component that performs the recited application of an indexing function. In that scenario, a person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek for reasons similar to those as discussed in Section IX.C, claim 3. (*See* Section IX.C.1.) 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 system (“applying information in the location query from the client”) to a hash function (“indexing function”) in order 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”). (See Section IX.C.1) As discussed for claim 7, Kahn’s hash function is an “indexing function.” (See Section IX.C.3.)

9. 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.”

128. The Kahn-Vingralek combination discloses and/or suggests claim 16. As discussed in Section IX.A, Kahn discloses claim 16. (See Section IX.A.10.) I have been asked to consider a scenario where 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”). In that scenario, a person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek for reasons similar to those as discussed in Section

IX.C, claim 3. (*See* Section IX.C.1.) As discussed in Section IX.C, 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. (*See* Section IX.C.8.)

10. Claim 17

- i) **[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:”**

129. I understand that this limitation is a preamble of the claim and I have been asked to assume that the preamble of claim 17 is limiting. Under such an assumption, Kahn in view of Vingralek discloses and/or suggests the limitations therein. Kahn discloses “a system having a plurality of location servers for storing and retrieving location information, wherein each of the plurality of location servers stores [a] unique set of location information of an aggregate set of location information.” 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”). (*See* Section 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 in Section IX.A, limitation [10.a]. (*See* Section IX.A.5.ii.)

130. 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 person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek.

131. As discussed in Section IX.C, claim 3, Vingralek and Kahn are in the same field of art, and a person of ordinary skill in the art would have been motivated to consider the teachings of Vingralek when implementing Kahn’s system/method. (*See* Section IX.C.1.) As discussed therein, Vingralek discloses a system where a file is organized into buckets spread across servers. (*See* Section IX.C.1; Ex. 1007, 4.) Vingralek employs a local load control at each server, preventing server overloading by redistributing buckets when a specified server utilization is reached. (Ex. 1007, 6.) Vingralek also implements an “explicit mapping of buckets onto servers,” allowing “additional flexibility to redistribute the load of an overloaded server” by, e.g., splitting bucket(s) of the server, migrating a bucket to a newly-acquired server, or migrating a bucket to an existing server. (*Id.*, 6-7.) Vingralek determines the utilization/performance capacity of the

existing servers and decide on splitting versus migrating buckets, and acquiring new servers versus redistributing across existing servers. (*Id.*, 7 (“The decisions about splitting versus migrating a bucket and acquiring a new server versus redistributing across existing servers depend on the overall utilization of (the performance capacity of) the existing servers. To make an intelligent decision, our approach makes an ‘educated guess’ of the total load of the entire system (i.e., all servers), based on a probabilistic model with sampling based information on bucket sizes.”).) Vingralek explains that a logically centralized server called a file advisor monitors/controls utilization/performance capacity. (*Id.*, 7, 12.)

132. 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) and determines whether to split/migrate buckets, or to acquire a new server, based on factors such as the number of keys added to the server since the overload message was sent, as well as

on whether other metrics such as global utilization have exceeded specified thresholds (*id.*). If a server ultimately reaches the panic load capacity, then 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.) Thus, Vingralek discloses scaling the capacity of a server by, e.g., increasing the effective capacity by transferring buckets of information to another server when certain thresholds (e.g., load capacity concepts) are reached.

133. Additionally, Vingralek discloses that its method of scaling “strives for the objective of ‘cost-conscious’ scalability, by aiming to minimize the number of servers...used for providing the necessary performance capacity” (*id.*, 6) and improves on previous systems because they does not consider “the cost of scaling up a distributed system..., and thus no attempt is made to minimize the number of servers across which a file is spread” (*id.*, 8). In particular, Vingralek discloses that its approach is scalable because new servers are acquired (e.g., as a way of managing the server load) only if server load metrics exceed certain thresholds. (*Id.*, 24.)

134. A person of ordinary skill in the art 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 reduce the cost of scaling up the distributed system. 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), consistent with Vingralek's teaching, a person of ordinary skill in the art would have been motivated 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 person of ordinary skill in the art 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. (*Id.*, 10-13.) Moreover, a person of ordinary skill in the art 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. (Ex.

1005, 9:23-27 (“**The handles used to uniquely identify digital objects are designed to be extensible and expandable to accommodate virtually any number of objects over many years.** The hashing mechanism provides an efficient and reliable implementation.”); 33:8-10 (“[T]he identifier’s namespace must be variable and unrestricted, **since the number of digital objects created may be expected to increase....**”).) A person of ordinary skill in the art would have been motivated to include features similar to Vingralek’s method of scaling server capacity when implementing Kahn’s system of handle servers, such that the combined system includes a method of scaling the capacity of a handle server in the system of handle servers (“[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”).

135. A person of ordinary skill in the art would have had a reasonable expectation of success combining the teachings of Kahn and Vingralek. As discussed above, both references are in the same field of art. (*See above.*) Moreover, Kahn itself suggests that the digital objects, and thus handle servers, may be scaled arbitrarily. (Ex. 1005, 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—implementing a means of scaling the

system of digital objects and handle servers. Additionally, the combination of Kahn and Vingralek 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. Accordingly, Kahn in view of Vingralek discloses and/or suggests this limitation.

ii) **[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;”**

136. Kahn discloses this limitation. 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”). (*See* Section 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.” (Ex. 1005, 26:8-10, FIG. 33 (showing a diagram of portions of a handle system).) Thus, a person of ordinary skill in the art

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

137. Kahn discloses this limitation. Consistent with as discussed for [1.a], Kahn discloses “storing location information...at a first location server.” (*See* Section IX.A.1.ii.) In particular, Kahn discloses storing one or more handles and associated pointers (collectively, “location information”) at a first handle server (“first location server”). (*Id.*)

138. 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. (*See* Section IX.C.10.ii.) A person of ordinary skill in the art 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.

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

139. Kahn discloses this limitation. 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. (Ex. 1005, 20:4-14 (An “originator”...is an entity that authorizes or validates a set of digital objects; it is responsible for each such digital object including making it available in the handles system...and defining terms and conditions for its use. Every digital object has an originator, which may be an individual or an organization. Originators may deposit and access the digital objects they authorize or validate and may authorize others to do so (this also includes the right to withdraw or modify the objects), subject to the procedures established by individual repositories.”), 37:4-6 (“An originator may control a naming authority, but there may be naming authorities that are not controlled by the originators.”).) 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 (“Naming authorities have the right to insert handle entries for handles they generate into the handle server system and to authorize others to do so.”), 39:34-40:1 (“Naming

authorities can choose to store the handles they create on any handle servers for which they have access permissions....”), 43:31-35 (“Handle servers provide a number of services, three of which are RESOLVE, INSERT, and DELETE. A party that is authorized to insert, delete and otherwise change handle entries for a particular naming authority is called a handle administrator.”), 45:10-21 (“However, if the location of the repository were to change, the handle server would have to be notified so it could make the corresponding changes....**The handle administrator may determine whether to store IP addresses or domain names or other information [i.e., pointers] in the handle server. The entries are typed and therefore one or more of the above information types may be provided by the administrator for retention in the handle server.**”).) 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”).

140. I have been asked to consider a scenario where Kahn does not explicitly disclose a handle server receiving both a handle and associated pointers for insertion. Under such a scenario, a person of ordinary skill in the art would have been motivated to include such a feature in Kahn. A person of ordinary skill in the art 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 (Ex. 1005, 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 person of ordinary skill in the art 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.

141. A person of ordinary skill in the art 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 (Ex. 1005, 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.

- v) **[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”**

142. Kahn discloses this limitation. 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”). (*See* Section IX.A.4.) A person of ordinary skill in the art would have thus understood that when a handle server (“first data location server”)⁶ stores the received handle

⁶ Kahn discloses a “data location server” under the Interpretation from Other Proceedings of “location server” for the same reasons as discussed for limitation [1.pre]. (Sections VIII, IX.A.1.ii.)

(“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”).

Accordingly, Kahn discloses this limitation.

- vi) **[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.”**

143. The Kahn-Vingralek combination discloses and/or suggests this limitation. 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”), a person of ordinary skill in the art would have been motivated 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. (Section IX.C.10.i.) Thus, the Kahn-Vingralek combination discloses and/or suggests transferring a portion of 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 method of claim 17, wherein the performance criterion comprises an amount of available persistent storage space in the first location server.”

144. The Kahn-Vingralek combination discloses and/or suggests this limitation. 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.” (Ex. 1007, 10, 21 (“[T]he panic load capacity of a server is expressed in terms of the maximum number of keys after which the server would start thrashing.”); Sections 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. (Ex. 1007, 21 (“[T]he record size and the key size were fixed throughout the simulation.”), 22 (displaying simulation parameter settings in which the panic load capacity is 11000 records, record size is 10 Kbytes, and key size is 100 Bytes).) Thus, a person of ordinary skill in the art 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 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.”**

145. The Kahn-Vingralek combination discloses and/or suggests this limitation. 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”). (*See* Section 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 person of ordinary skill in the art would have been motivated 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. Kahn in View of Vingralek and Krasner Discloses and/or Suggests Claim 11

1. Claim 11: "The system of claim 10, wherein the location information comprises geographic location information."

146. The Kahn-Vingralek combination in view of Krasner discloses and/or suggests claim 11. As discussed in Section IX.C, claims 3 and 10, a person of ordinary skill in the art would have been motivated to combine teachings of Kahn and Vingralek. (Sections IX.C.1, IX.C.4.) For similar reasons to those discussed for claim 2, a person of ordinary skill in the art would have been motivated to combine the Kahn-Vingralek combination with Krasner. (Section 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") of a mobile GPS unit that may be provided to a client system. (See Section IX.B.1.) Accordingly, the Kahn-Vingralek-Krasner combination discloses and/or suggests claim 11.

E. Kahn in View of Vingralek and Sato Discloses and/or Suggests Claims 24-26

1. Claim 24: “The method of claim 17, wherein the performance criterion comprises a transaction rate limit.”

147. Kahn in view of Vingralek and Sato discloses and/or suggests claim 24. As discussed for limitations [17.pre] and [17.e], the Kahn-Vingralek combination discloses and/or suggests a method of scaling capacity of a location server, and transferring certain identifiers and associated locations to a second data location server when a performance criterion of the first location server reaches a predetermined performance limit. (Sections IX.C.10.i, IX.C.10.vi.) While the Kahn-Vingralek combination does not disclose scaling a location server’s capability where the performance criterion comprises a transaction rate limit, a person of ordinary skill in the art would have been motivated to implement this feature in view of Sato.

148. Like Vingralek, Sato describes a “client-server system built on a distributed system” where the client “issue[s] a service request to a specific server and receive[s] a service.” (Ex. 1011, ¶[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] (“One method to efficiently investigate the existence of a server that provides the service requested

by a client and to efficiently obtain information for establishing the communicating means with the server is to use a server that services location information of various servers (hereinafter referred to as a service location server).”) Sato further discloses a “server partitioning method that can partition the service range and manage location information in a distributed manner when the load on the service location server exceeds a predetermined threshold.” (*Id.*, ¶[0005].) Thus, Sato is similar to Vingralek, which discloses a method of migrating information from one server to another when the first server reaches a performance criterion threshold. (Ex. 1007, 5-8, 13.) A person of ordinary skill in the art would have therefore been motivated to consider the teachings of Sato when implementing the Kahn-Vingralek combination.

149. Sato additionally discloses that each “service location server” includes a “load measuring means” measuring “service load information” of the server (Ex. 1011, ¶[0009]), where the measured “service load information” includes “the maximum number of service executions per unit time...after starting the service of the server” (*id.*, ¶[0012]). “Whether or not 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”).

150. A person of ordinary skill in the art would have been motivated to include features similar to those disclosed by Sato when implementing the Kahn-Vingralek combination. For example, Vingralek assumes that the “server instruction rate” (i.e., server processing speed) is constant. (Ex. 1007, 22 (noting that the server instruction rate is held at a constant 10 MIPS (million instructions per second))).) While Vingralek generally describes a homogenous system (*id.*, 21), its approach is “applicable also to heterogeneous servers where servers...may have different load capacities” (*id.*, 26). Thus, a person of ordinary skill in the art would have been motivated to modify the Kahn-Vingralek system to account for heterogeneous servers where the server instruction rate (processing speed) may vary 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.” (Ex. 1011, ¶[0073].) Additionally, Sato’s method would have been beneficial because it would have allowed implementing a partitioning method in a system of heterogeneous servers, where processing speed may vary. Consistent with the above and in such a combined system, for example, 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.

151. A person of ordinary skill in the art would have also had a reasonable expectation of success combining the teachings of Kahn, Vingralek, and Sato. As discussed above, both references are in the same field of art. (*See above.*) 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. Additionally, the combination of Kahn, Vingralek, and Sato 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.

2. Claim 25: “The method of claim 17, wherein the transaction rate limit comprises a processor speed limit.”

152. The Kahn-Vingralek-Sato combination discloses and/or suggests claim 25. As discussed for limitation [1.c], each handle server may be a computer, and thus a person of ordinary skill in the art would have understood that a handle server includes a “processor.” (*See* Section IX.A.1.iv.) As discussed for claim 24, the performance criterion is a maximum service *execution* rate (“transaction rate limit”). (*See* Section IX.E.1.) Thus, a person of ordinary skill in the art 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 method of claim 17, wherein the transaction rate limit comprises a network connection bandwidth limit.”

153. The Kahn-Vingralek-Sato combination discloses and/or suggests claim 26. Vingralek discloses the **performance** of the distributed network is “characterized by the network latency *NL* (i.e., the time required to process each packet, independently of the packet size) and the **bandwidth** *BW* (i.e., the

transmission time that is needed to send one byte through the network).” (Ex. 1007, 22.) A person of ordinary skill in the art would have understood that the network bandwidth is of a finite value/number. Thus, Vingralek discloses “a network connection bandwidth limit.” A person of ordinary skill in the art 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 the maximum service execution rate of the servers. (*See* Section IX.C.10.i.)

154. In view of the above teachings, a person of ordinary skill in the art, when implementing the Kahn-Vingralek-Sato combination would have 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” (as recited in limitation [17.e]), where the “performance criterion” is “the transaction rate limit compris[ing] a network connection bandwidth limit” (as recited in claim 26). (*See* Section IX.C.10.vi.) For example, 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. (Ex. 1007, 26; Section IX.E.1.) In that scenario, a person of ordinary skill in the art 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. (Section IX.C.10.vi.) Such a process would have reduced the load of the first handle server, and increased the utilization of the second handle server, consistent with the goal of Vingralek that “strives for the objective of ‘cost-conscious’ scalability, by aiming to minimize the number of servers that are used for providing the necessary performance capacity.” (Ex. 1007, 6.)

155. A person of ordinary skill in the art would have also had a reasonable expectation of success in implementing the above-discussed feature in the Kahn-Vingralek-Sato combination. Indeed, it would have involved modifying the Kahn-Vingralek-Sato system in a manner that Vingralek suggests is feasible—implementing methods of partitioning servers in a heterogeneous system. Additionally, 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, as discussed above.

X. CONCLUSION

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

Dated: June 28, 2024

By:  _____

Sandeep Chatterjee