

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner

v.

KOVE IO, INC.,
Patent Owner

Patent No. 7,103,640

**DECLARATION OF DR. SANDEEP CHATTERJEE
IN SUPPORT OF PETITION FOR INTER PARTES REVIEW
OF U.S. PATENT NO. 7,103,640**

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 '640 PATENT	16
	A. Description	16
VII.	OVERVIEW OF THE PRIOR ART	19
	A. Kahn	19
	B. Vingralek	25
VIII.	CLAIM CONSTRUCTION	28
	A. "identifier"	29
	B. "location information"	29
	C. "location server"	29
	D. "client"	30
	E. "plurality of location servers"	30
IX.	THE PRIOR ART DISCLOSES AND/OR SUGGESTS ALL RECITED FEATURES OF CLAIMS 10-24 OF THE '640 PATENT	33
	A. Kahn Discloses Claims 10-13, 17-19, and 24	33
	1. Claim 10	33
	2. Claim 11	72
	3. Claim 12	74
	4. Claim 13	77

Declaration of Dr. Sandeep Chatterjee
U.S. Patent No. 7,103,640

5.	Claim 17	78
6.	Claim 18	84
7.	Claim 19	92
8.	Claim 24	93
B.	Kahn in View of Vingralek Discloses and/or Suggests Claims	
	10-24	96
1.	Claim 10	96
2.	Claim 11	108
3.	Claim 12	109
4.	Claim 13	111
5.	Claim 14	111
6.	Claim 15	112
7.	Claim 16	112
8.	Claim 17	114
9.	Claim 18	116
10.	Claim 19	117
11.	Claim 20	118
12.	Claim 21	118
13.	Claim 22	119
14.	Claim 23	119
15.	Claim 24	120
X.	CONCLUSION	120

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. 7,103,640 (“the ’640 patent”) (Ex. 1001).¹ I have been asked to consider whether prior art references disclose or suggest the features recited in claims 10-24 (“the challenged claims”) of the ’640 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 ’640 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,

Declaration of Dr. Sandeep Chatterjee
U.S. Patent No. 7,103,640

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:

Ex. 1001	U.S. Patent No. 7,103,640 to Overton et al. (“the ’640 patent”)
Ex. 1003	Curriculum Vitae of Sandeep Chatterjee, Ph.D.
Ex. 1004	Prosecution History of U.S. Patent No. 7,103,640
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. 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 '640 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 '640 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 '640 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 '640 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 '640 patent as of around July 8, 1998. Considering the '640 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 '640 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 '640 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 '640 patent. I rely on, and incorporate as applicable (even if not expressly mentioned below in Section IX), the following

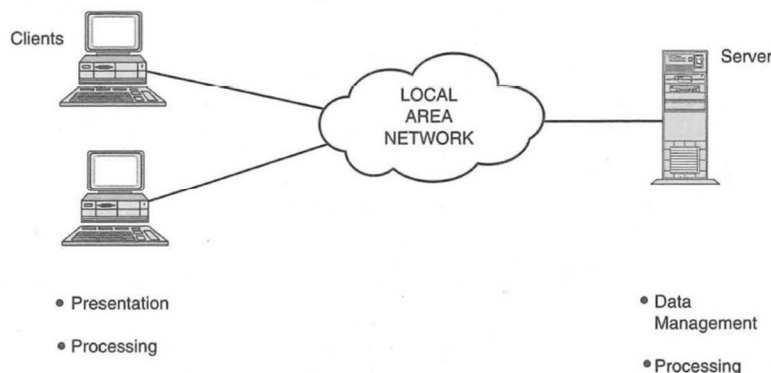
disclosures and opinions to support my opinions in this Declaration, including those opinions relating to how the prior art discloses and/or suggests the challenged claims of the '640 and how and why a person of ordinary skill in the art would have been motivated to consider and combine the disclosures 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, 15:1-22 (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.**”).)

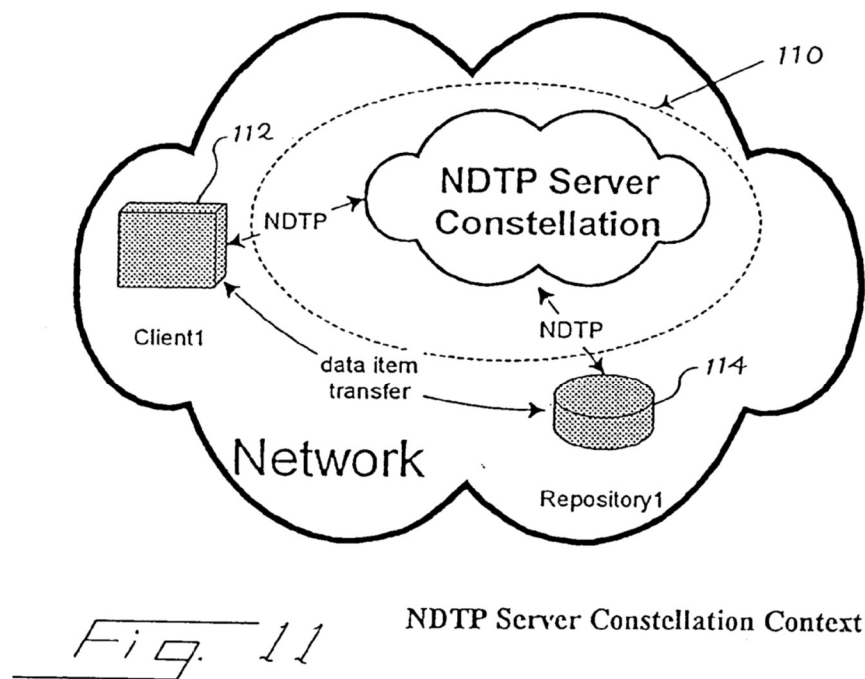
27. Hash functions are used in many applications, but as an example, a hash function can be used to locate a tax record associated with a social security number (SSN) within a distributed system, where the SSNs are not stored numerically. (*Id.*, 209 (“For example, suppose that the California tax records 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 '640 PATENT

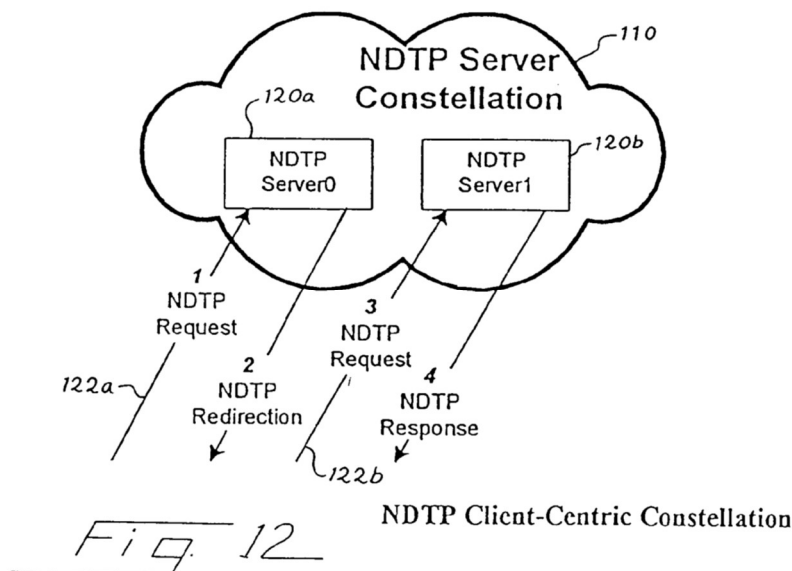
A. Description

29. The '640 patent describes storing and retrieving information across a network (Ex. 1001, 1:26-30, 1:56-58) and a network distributed tracking protocol (NDTP) (*id.*, 2:12-14, 4:25-36). Figure 11 shows an NDTP server constellation where a client consults servers that return strings in response to a client request. (*Id.*, 16:53-56.)



(*Id.*, FIG. 11.)

30. Server constellations have two organizational approaches: client-centric (shown in Figure 12) and server-centric (shown in Figure 13). In the client-centric approach, processing emphasis is toward clients rather than toward servers. (*Id.*, 18:11-13).



(*Id.*, FIG. 12.) In contrast, in the server-centric approach, server processing is emphasized. (Ex. 1001, 18:37-41.)

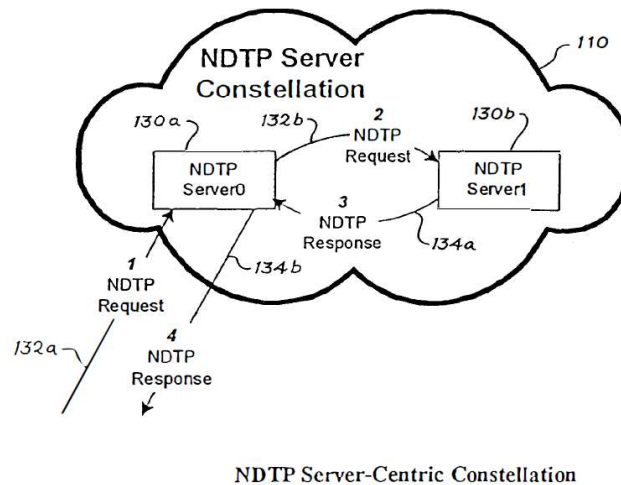


Fig. 13

(*Id.*, FIG. 13.)

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, 2:12-44.) 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....”).)

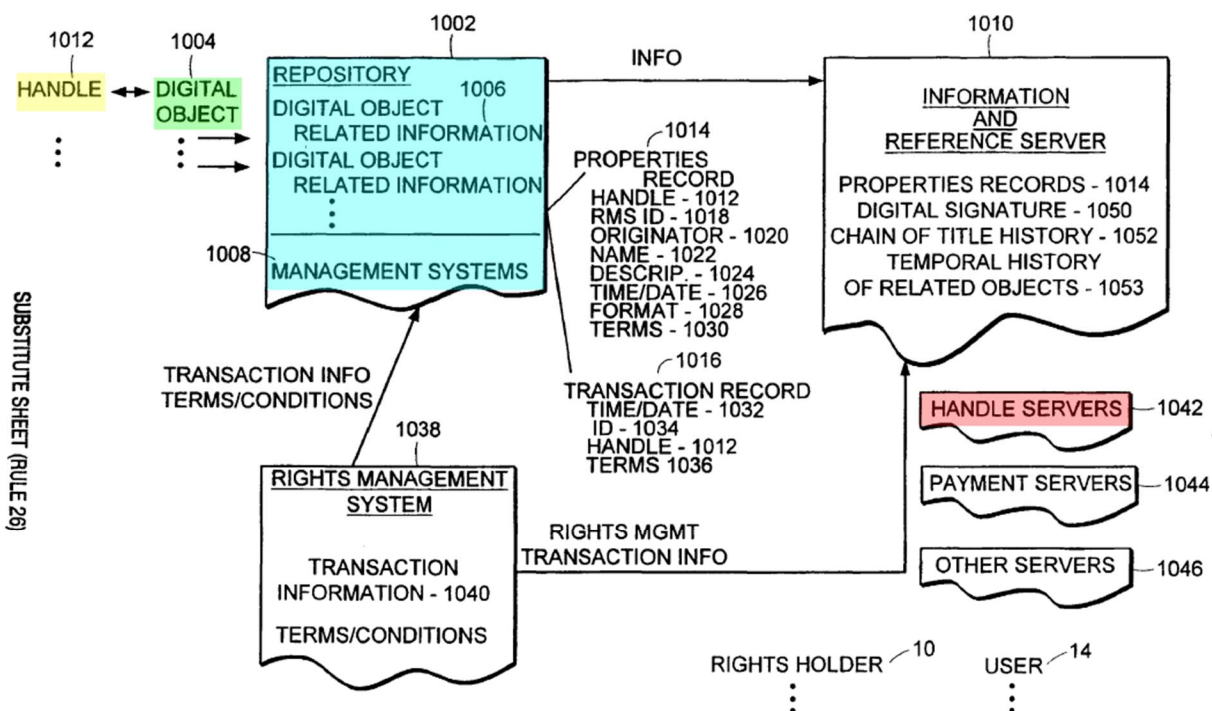


FIG. 1

(*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.)

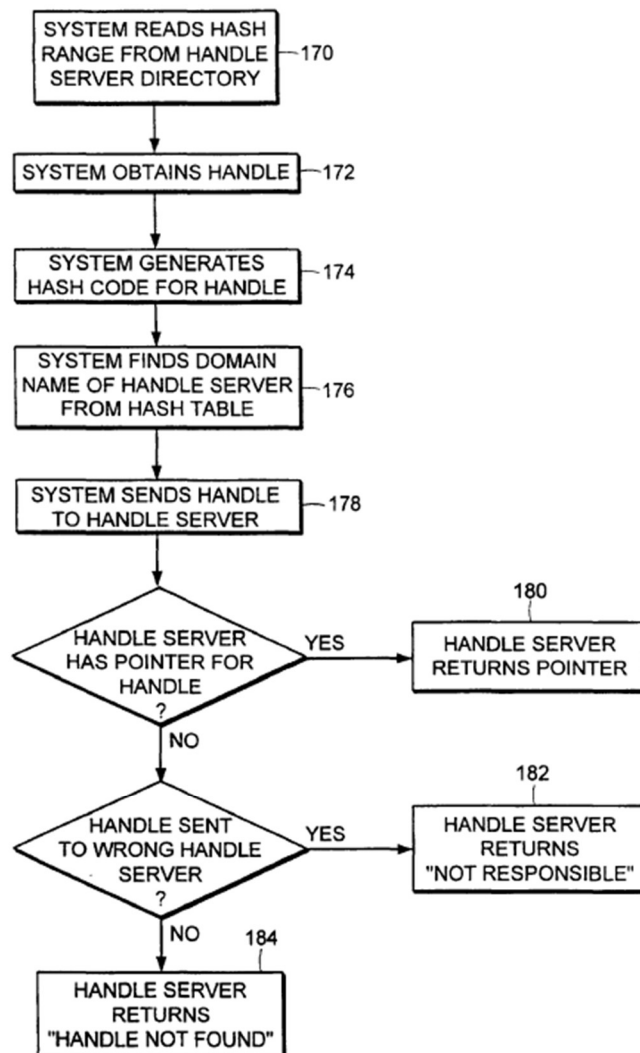


FIG. 8
SUBSTITUTE SHEET (RULE 26)

(*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.”).)

VIII. CLAIM CONSTRUCTION

39. 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 ’640 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”

40. I have been asked to interpret the claimed “identification string” and “identifier string” recited in claims of the ’640 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.”

B. “location information”

41. I have been asked to interpret the claimed “location information” recited in claims of the ’640 patent to mean:

“information pertaining to one or more locations of data and/or the identities of one or more location servers.”

C. “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.)

42. I have been asked to interpret the claimed “location server” recited in claims of the ’640 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”

43. I have been asked to interpret the claimed “client” recited in claims of the ’640 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”

44. 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 ’640 patent to mean:

“location servers in non-hierarchical structures.”

45. 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.”

'640 Patent Terms	Interpretation from Other Proceedings
“identification string”; “identifier string”	“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”	“information pertaining to one or more locations of data and/or the identities of one or more location servers”
“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 initiates update or lookup of identifier/location mappings from a location server with location request messages”

Declaration of Dr. Sandeep Chatterjee
U.S. Patent No. 7,103,640

'640 Patent Terms	Interpretation from Other Proceedings
“plurality of location servers”; “plurality of data location servers”	“location servers in non-hierarchical structures”

IX. THE PRIOR ART DISCLOSES AND/OR SUGGESTS ALL RECITED FEATURES OF CLAIMS 10-24 OF THE '640 PATENT

46. In my opinion, the prior art discussed below discloses and/or suggests the limitations of claims 10-24.³

A. Kahn Discloses Claims 10-13, 17-19, and 24

1. Claim 10

- i) [10.pre] “A method for retrieving data location information for data stored in a distributed network, comprising the steps of:”**

47. 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 the limitations therein. To begin, 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...in a

³ 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.

network is presented. The objects are stored at locations...accessible in the network....Other aspects include providing multiple servers...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.”), 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.”).)

Multiple handle servers in the network each may accept handles of digital objects and return corresponding pointers that indicate the digital objects’ locations 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.”), 7:2-20 (“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.”.) For example, Kahn discloses with reference to Figure 1 an exemplary system that includes a repository 1002 (blue) for storing digital objects 1004 (green) in the network, where the digital objects are identified by their handle 1012 (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 (red). (*Id.*, Abstract, 7:19-20 (“...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....”).)

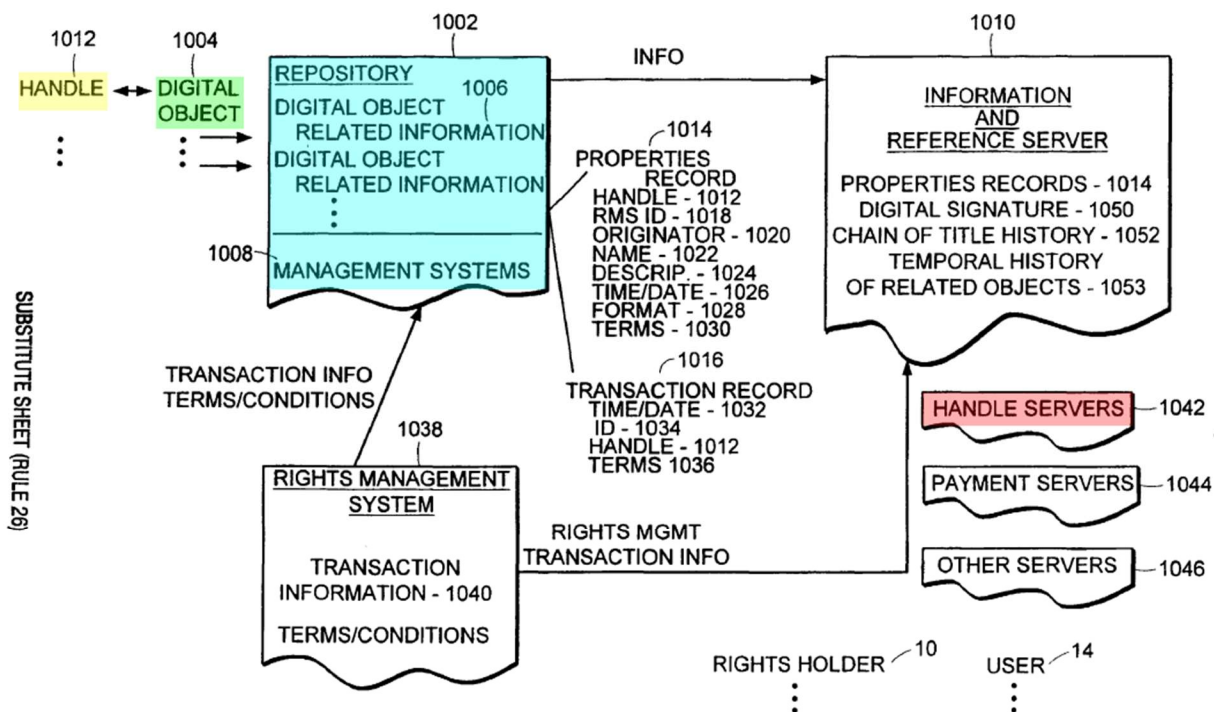


FIG. 1

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

48. Kahn discloses that digital objects (including “data”) are “stored in a distributed network.” A digital object is represented by a sequence of bits or digits and may include conventional digital representations of works (e.g., books, images, sounds, software), and each digital object is identified by a unique handle. (*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.”), 1:16-20 (“Thus digital objects may include conventional digital 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.”).)

A person of ordinary skill in the art would have understood that sequences of bits/digits associated with each digital object are “data,” as claimed. 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.)

Kahn discloses that a digital object is stored in a repository in a distributed network, such as the Internet, where the network does not have a central nodal point.⁴ (*Id.*, Abstract (“**The [digital] objects are stored at locations...accessible in the network.**”), 2:25-30 (“digital objects available on networks (e.g., **Internet**)”), 6:9-35 (“Copies of the digital objects are stored in a repository....The

⁴ Kahn’s distributed network (e.g., Internet) is consistent with the ’640 patent’s disclosure associated with a distributed network. (Ex. 1001, 1:56-65 (“A distributed data collection is a system where data is stored and retrieved among multiple machines connected by a network.”), 4:49-57 (describing a network distributed across Internet machines).)

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’...is a digital storage system into which digital objects...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).”).)

49. Kahn explains that one or more pointers (“data location information”) returned by a handle server indicate the location of a stored digital object in the distributed network. (*Id.*, Abstract (“[M]ultiple 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....**”), 3:21-22 (“[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....”).) Kahn also discloses the claimed “data location information” under the Interpretation from Other Proceedings for “location information,” i.e., “information pertaining to one or more locations of data and/or the identities of one or more location servers,” because as discussed above, the pointers provide information pertaining to one or more locations of digital objects (including “data”). (Section VIII.)

50. Kahn discloses “retrieving data location information for data” stored in the distributed network. Kahn discloses with reference to Figure 8 (below) a client system retrieving one or more pointers (“data location information”) for digital objects (including “data”). (Ex. 1005, 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.)

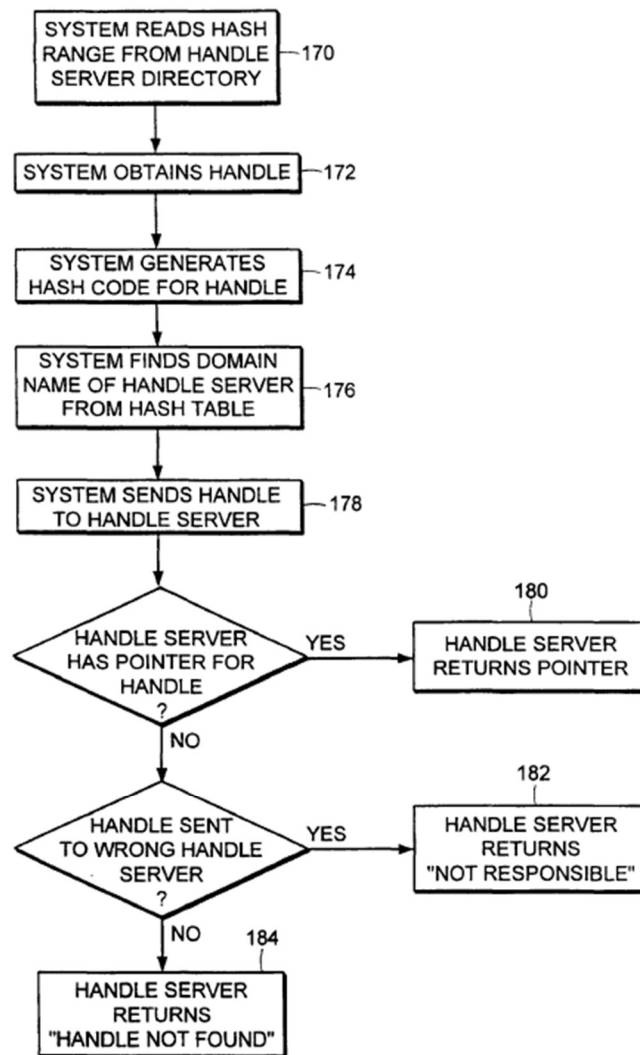


FIG. 8
SUBSTITUTE SHEET (RULE 26)

(*Id.*, FIG. 8.) As shown in Figure 8 step 170, a client first, e.g., at initiation, downloads from a handle directory server a hash range/handle-server-domain-name table (also referred to as a hash code/handle server table, *see* Figure 7 below) that associates hash ranges with handle server domain names. (*Id.*, 46:22-47:5,

FIG. 7.) The table allows the client system to determine the location of a handle server that contains 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....”).)

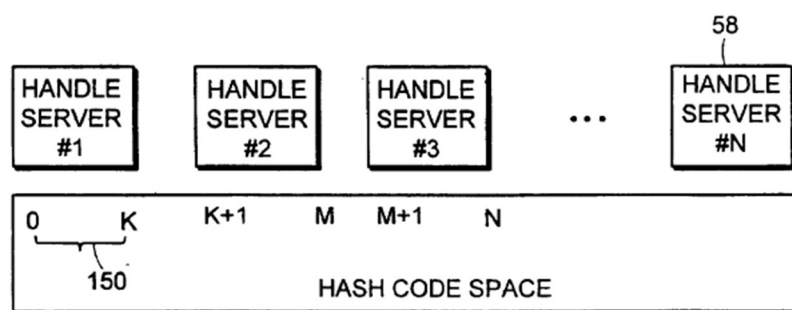


FIG. 7

149 ↗

(*Id.*, FIG. 7.) For instance, as shown in Figure 7 above, each handle server is assigned a range of hash codes/values 150. (*Id.*, 46:22-24; *see also id.*, FIG. 7.) The client uses a generated hash code/value (further discussed below) 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 pointer for locating the digital object associated with the handle. (*Id.*, 47:1-11, 73:17-26.)

51. Returning to Figure 8, after downloading the hash range/handle-server-domain-name table in step 170, the client in step 172 receives a handle from a user's request (by, e.g., a bibliographic search or scanning a barcode) 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...").) Subsequently, the client system "generates the hash code for the handle using a predetermined hashing algorithm," 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,” and sends the handle to the appropriate handle server in steps 174, 176, and 178 of Figure 8. (*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)....”).)

52. Next, the handle server verifies if it does or, e.g., in the event of a dynamic change to the table downloaded to the client, does not contain the handle and its associated pointers. (*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 indeed located in the queried handle server, one or more pointers for locating the digital object stored in the distributed network are returned to the client system in step 180 (“retrieving data location information for data stored in a distributed network”). (*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 system then uses the retrieved pointers to obtain the digital object stored in a repository in the distributed network. (*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.”).) Accordingly, Kahn discloses this limitation.

- ii) [10.a] **“a) receiving at a first client a data query for retrieving data associated with an identification string, wherein the data is stored at a data repository in the distributed network and wherein a location string associated with the identification string of the data is stored in at least one of a plurality of data location servers;”**

53. Kahn discloses this limitation. **First**, Kahn discloses “data associated with an identification string.” Kahn discloses that each digital object (including “data”) is associated with a unique handle (“identification string”)⁵ identifying that digital object, where the handle comprises a unique character string. (Ex. 1005, 7:4-5 (“The handles comprise unique character strings associated with the servers which generated them.”).) 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

⁵ Kahn’s handle is consistent with the ’640 patent’s disclosure associated with the identification string. (Ex. 1001, 2:16-18 (“identification strings for specifying the identity of an entity in the distributed data collection”); *see also id.*, 2:49-51, 2:52-53.)

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....**”); Section IX.A.1.i.)

54. Kahn’s handle is an “identification string” under the Interpretation from Other Proceedings for related term “identifier,” 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* 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....”).) Kahn also explains that a handle may be associated with zero or more pointers (i.e., zero or more data location specifiers) in a handle server (i.e., a location server, as interpreted in the Other Proceedings and discussed below), and the pointer specifies the location of the digital object (including “data”). (*Id.*, 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....”); *see also id.*, 7:25-27 (“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....”).)

55. **Second**, Kahn discloses “receiving at a first client a data query for retrieving data associated with an identification string.” For example, Kahn discloses the client system (“first client”) receiving a request (“data query”) from a user for retrieving a digital object (including “data”) associated with a handle (“identification string”). A user may request a search for a digital object by using

an object's handle as input to the client system, and the client then performs various processes to locate the digital object associated with the object's handle. (*Id.*, 73:8-14 (“A user (or more likely, his software agent) obtains a handle 743 for an object (742). 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.**”), 73:14-74:32 (explaining that “[o]nce the handle is obtained” from the user by the requesting client system, the system proceeds with the search for the digital object); *see also* above Section IX.A.1.i. (describing client's processes to locate digital object).) Specifically, the client system generates the hash code for the handle using a hashing algorithm, and uses the hash code to index into the hash range/handle-server-domain-name table to determine the domain name of the handle server that may contain a pointer to the digital object. (Ex. 1005, 47:1-5; *see also id.*, 73:17-22; *see* above Section IX.A.1.i.) The client sends a request including the handle to the handle server, and if the handle server contains the handle and associated pointers, the pointers are returned to the client for it to locate and retrieve the digital object associated with the handle. (Ex. 1005, 47:5-11 (“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).**”), 74:23-26 (“For each pointer returned by the handle server, the requesting system uses the pointer to attempt to obtain a copy of the digital object 762.”).)

56. Kahn discloses the claimed “first client” under the Interpretation from Other Proceedings for “client,” 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, 4:11-14.) Kahn discloses a user providing a handle to a client system (“first 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 client system’s user interface may be “in the form of software running on a workstation.” (*Id.*, 26:28-34 (“As seen in Figure 2, an 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 (displaying the UA on the client system).)

Kahn further 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 user’s computer or client system (“first client”) is either a hardware and/or software component attached to a network. As discussed above and further discussed for limitation [10.b], the client system initiates a lookup of handle-to-pointer mappings (mappings of “identification string[s],” i.e., identifiers, and “location string[s],” i.e., location information, respectively) from a handle server (i.e., location server) storing such mappings, where initiation occurs by sending a request (i.e., location request message) to the handle server to retrieve the location information associated with a digital object. (*See* Section IX.A.1.iii.)

57. **Third**, Kahn discloses “wherein the data is stored at a data repository in the distributed network.” As discussed for limitation [10.pre], digital objects (including “data”) are stored in a repository (“data repository”) in a “distributed” network, where the network is comprised of the repository and other servers. (Ex. 1005, Abstract (“The [digital] objects are stored at [repositories] 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’ ...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).”); *see* Section IX.A.1.i.)

58. **Fourth**, Kahn discloses “a location string associated with the identification string of the data.” As discussed above, a handle uniquely identifies a digital object, which includes data (“identification string of the data”), and Kahn further discloses that a pointer (“location string”), which indicates a digital object’s location, is associated with a digital object’s handle (“identification string of the data”). (Ex. 1005, 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.”).) Moreover, a handle is used to obtain a list of pointers to the location of a digital object, where each pointer includes the domain name of a repository. (*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.”), 6:30-35 (“**The handle is used to obtain a pointer to the network location...of the digital object.**”), 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}.”).) Additionally, for reasons discussed above, Kahn’s pointer discloses the “location string” (or similar terms) as described in the ’640 patent. (Ex. 1001, 4:9-11.)

59. **Fifth**, Kahn discloses “wherein a location string...is stored in at least one of a plurality of data location servers.” Kahn discloses multiple handle servers and each handle server “holds pointers associated with a subset of all handles.” (*Id.*, 37:31-33 (“Handle servers have the following characteristics: a handle server holds pointers associated with a subset of all handles....”); *see also id.*, 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’)...”).) Because the handle servers store pointers for locating digital objects (including “data”), Kahn’s handle servers are the claimed “data location servers.” (*Id.*, 7:5-20 (“A handle server, accessible on the network, provides the pointer in response to the presentation of a handle....[There are] 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....”).)

60. Kahn also discloses the claimed “data location server[]” under the Interpretation from Other Proceedings for “location server,” 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.” (*See* above Section VIII; Ex. 1001, 4:14-18.) 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.”.) Each handle server maintains mappings between handles (i.e., identifiers) and pointers (i.e., locations). (*See* Section IX.A.1.i; Ex. 1005, 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....”), 21:6-10 (“[M]ultiple servers...provide...handle-to-pointer translation services (called ‘handle servers’)....”).) Further, Kahn discloses that the handle server returns identifier/location mappings in response to a request from the client (“first client”) for location information associated with a handle (i.e., in response to location request messages from a client). For example, as discussed below for limitation [10.b], 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. (*See* Section IX.A.1.iii.) 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 ’640 patent’s disclosure. (Ex. 1001, 2:36-40, 2:60-65, 6:63-67, 16:53-56.) 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, 6:46-49.) Thus, Kahn discloses the claimed “data location server[]” under the Interpretation from Other Proceedings for “location server.”

61. Further, Kahn discloses the claimed “plurality of data 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 “data 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. Accordingly, Kahn discloses this limitation.

- iii) **[10.b] “b) transmitting a data location request from the first client to a first data location server to retrieve the location string associated with the identification string in the data query, the data location request including the identification string;”**

62. Kahn discloses this limitation. **First**, Kahn discloses “the location string associated with the identification string in the data query.” As discussed for limitation [10.a], Kahn discloses one or more pointers (“location string”) are associated with a handle (“identification string”), where the handle is included in the request (“data query”) to retrieve a digital object. (*See* Section IX.A.1.ii.)

63. **Second**, Kahn discloses “transmitting a data location request from the first client to a first data location server to retrieve the location string..., the data location request including the identification string.” Kahn discloses that the client system (“first client”) transmits a handle query in the form of a request pointer information UDP packet (“data location request”), which includes the handle (“identification string”), to the handle server (“first data location server”) to retrieve one or more pointers (“location string”) for locating the requested digital object. (Ex. 1005, 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 (step 180).”); *see also id.*, 7:5-7 (“A handle server, accessible on the network, provides the pointer in response to presentation of a handle.”), 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....”), 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.”), 73:32-34 (“The handle server verifies that the handle falls within the set of handles for whose hash values it is responsible....”); Section IX.A.1.ii.) Accordingly, Kahn discloses this limitation.

- iv) [10.c] “c) if the first data location server does not possess the location string, transmitting a redirect message to the first client, the redirect message containing information for use by the first client to calculate a location of a second data location server, wherein the second data location server contains the location string;”

64. Kahn discloses this limitation. As an initial matter, 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.

65. **First**, under the Applied Interpretation, Kahn discloses “if the first data location server does not possess the location string, transmitting a redirect message to the first client.” Kahn discloses transmitting a not-responsible-for-handle message (“redirect message”) to the client system (“first client”) in the form of an invalid-handle-server-selected response UDP packet if the handle server, i.e., a “wrong handle server,” does not contain the handle and its associated pointers (“if the first data location server does not possess the location string”). (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.)

66. **Second**, under the Applied Interpretation, Kahn discloses “the redirect message containing information for use by the first client to calculate a location of a second data location server.” If the handle was sent to an incorrect handle server, a not-responsible-for-handle message (“redirect message”) is transmitted to the client, informing the client to download an updated table from the handle server directory and recalculate the location of a correct handle server containing the handle (“to calculate a location of a second data location server”). (*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.”.) If a cache miss constitutes a “redirect message,” as in the Applied Interpretation, Kahn’s not-responsible-for-handle-message likewise discloses the claimed “redirect message” that contains “information for use by the first client to calculate a location of a second data location server” because it informs the client to download an updated table to determine the location of a correct handle server.

67. **Third**, under the Applied Interpretation, Kahn discloses “wherein the second data location server contains the location string.” Kahn discloses that when attempting to find the correct handle server which contains the pointers (“second data location server contains the location string”), the client system reattempts calculating a location of another handle server until it finds the correct handle server.⁶ (*Id.*, 47:13-18 (“If the handle was sent to the wrong handle server, it

⁶ While Kahn discloses that the subsequent attempt(s) occur a certain number of times before the system gives up, a person of ordinary skill in the art would have recognized that this number of attempts would have been set sufficiently large such

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 from the handle server directory again and attempt the mapping again.”.) A person of ordinary skill in the art would have recognized that for any client request for a stored digital object that takes more than one attempt to find a correct handle server, there is some first handle server (“first data location server”) that receives a request but does not have the requested pointers and thus sends a redirect message to the client system with information the client uses to calculate the location of some second handle server that does have the requested pointers (“second data location server contains the location string”).

that if the digital object actually exists in the system, the pointers for that digital object may be found by repeatedly reattempting the request. (Ex. 1005, 47:18-21 (“The requester will determine how many times to try before giving up....”).) In other words, the allowed number of attempts would have been set sufficiently high such that running out of attempts would indicate that the digital object does not exist.

68. Kahn also discloses the claimed “redirect message” even if the Applied Interpretation is not applied. As discussed above, the hash range/handle-server-domain-name table (“redirect message”) is downloaded by the client/requesting system (“transmitted...to the first client”) in response to 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 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 transmitted in/by the handle system, e.g., transmitted by the handle server directory of the handle system, 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 transmission of the Kahn’s “redirect

message” occurs whenever the not-responsible-for-handle message is sent, i.e., “if the first data location server does not possess the location string.” The message, including the handle-server-domain-name table, “contain[s] information for use by the first client to calculate a location of a second data location server.” As explained below for limitation [10.d], the transmitted/updated table associates hash ranges with handle server domain names, and allows the client system to determine/calculate the location of the correct handle server (“second data location server”) containing a pointer (“location string”) for retrieving the digital object associated with the handle. (*See* below Section IX.A.1.v; *see also* Section IX.A.1.i.) Thus, Kahn discloses this limitation even if the Applied Interpretation is not applied.

v) [10.d] “d) calculating the location of the second data location server at the first client; and”

69. Kahn discloses this limitation. For example, Kahn discloses downloading an updated hash range/handle-server-domain-name table and re-applying a hash algorithm/function to a handle to determine/calculate the location of the correct handle server (“calculating the location of a second data location server”) at the client system (“at the first client”). As discussed for limitations [10.pre] and [10.c], if a client system receives a not-responsible-for-handle message (“redirect message” under the Applied Interpretation), the client system downloads an updated hash range/handle-server-domain-name table containing hash ranges associated with the locations of handle servers, which contain the handle and its associated pointers. (See Sections IX.A.1.i, IX.A.1.iv.) Also as discussed for limitation [10.c], the “redirect message” is also disclosed by Kahn’s updated hash range/handle-server-domain-name table being transmitted to the client. (See Section IX.A.1.iv.) The client system then repeats the prior steps using the refreshed table. (Section IX.A.1.iv; Ex. 1005, 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.*, 74:5-12 ("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. This will typically be needed only if the table has changed between the time the table was downloaded and the actual request was made.'').)

70. This repetition of prior steps includes the client system calculating a hash code/value by applying a hash algorithm to the handle at the client system, and using the hash code/value to index into the updated table's hash range to determine the domain name/network address of the correct handle server containing the handle and its associated pointer. (Ex. 1005, 38:11-14 ("The directory servers will maintain a table of network addresses of handle servers (generally, each handle server will contain such a directory)."), 47:1-5 ("The [client] system...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)."), 73:17-20 ("Once the handle is obtained, the system that retrieves the

Declaration of Dr. Sandeep Chatterjee
U.S. Patent No. 7,103,640

object ‘hashes’ the object’s handle and uses this hashed value to perform a table lookup in the hash code/handle server table 744.”); *see also* Section IX.A.3 (further describing Kahn’s hash function).) Accordingly, Kahn discloses this limitation.

vi) [10.e] “e) transmitting the data query from the first client to the second data location server.”

71. Kahn discloses this limitation. Kahn discloses transmitting, from the client system (“first client”) to the correct handle server (“second data location server”), the request (“data query”) to retrieve a digital object. (*See also* Section IX.A.1.ii.) As discussed for limitation [10.d], after receiving a not-responsible-for-handle message (“redirect message” under the Applied Interpretation), or alternatively the updated hash range/handle-server-domain-name table itself (“redirect message”), the client system determines the location of a correct handle server (“second data location server”) using the updated table. (Section IX.A.1.v.) After locating the correct handle server, the request (“data query”) to retrieve a digital object is transmitted from the client system to the correct handle server. (Ex. 1005, 47:13-18 (“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 from the handle server directory again and attempt the mapping again.”); 74:5-26 (“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, and the requesting system repeats prior steps. []). 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.”); *see* Section IX.A.1.v.) In particular, the request (“data query”) to retrieve the digital object includes sending the handle to the correct handle server (“second data location server”) to return an associated pointer to locate and retrieve the digital object. (Ex. 1005, 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 (step 180).”).)

2. Claim 11: “The method of claim 10, wherein transmitting the redirect message comprises transmitting a data location server table to the first client.”

72. Kahn discloses this limitation. For example, Kahn discloses transmitting the not-responsible-for-handle-message (“redirect message” under the Applied Interpretation) comprises transmitting a hash range/handle-server-domain-name table (“data location server table”) to the client system (“first client”). As discussed for limitations [10.c]-[10.d], Kahn discloses that the result of the client system receiving such a “redirect message” is the subsequent transmission of the updated table to the client system. (*Id.*) (*See* Sections IX.A.1.iv-v; Ex. 1005, 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 Figure 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.*, 74:5-12 (“**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, 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.”).) Thus, transmitting the “redirect message” comprises transmitting the updated table to allow the client system to calculate the correct handle server (“second data location server” as discussed for limitations [10.c]-[10.e]). (*See* Sections IX.A.1.iv-vi.)

73. Also as discussed for limitation [10.c], the “redirect message” is also disclosed by the transmission of Kahn’s updated hash range/handle-server-domain-name table to the client. (*See* Section IX.A.1.iv.) As discussed therein, transmitting the updated table to the client system discloses “transmitting the redirect message,” which comprises “transmitting a data location server table.” (*Id.*) Accordingly, Kahn discloses this limitation.

3. **Claim 12: “The method of claim 11, wherein calculating the location of the second data location server comprises calculating the location of the second data location server with a function commonly known to the first data location server and the first client and based on the data location server table and the identification string.”**

74. Kahn discloses this limitation. **First**, Kahn discloses “wherein calculating the location of the second data location server comprises calculating the location of the second data location server with a function...and based on the data location server table and the identification string.” As discussed for limitation [10.d], Kahn discloses applying a hash algorithm (“function”) to a handle to determine/calculate the location of a correct handle server (“calculating the location of the second data location server”) to obtain one or more pointers associated with the handle. (*See* Section IX.A.1.v.) As further discussed therein, a hash algorithm is applied by the client system (“first client”) to the handle (“identification string”), resulting in a hash code/value used to index into the updated hash range/handle-server-domain-name table (“data location server table”) in order to determine/calculate the location of the correct handle server (“calculating the location of the second data location server...and based on the data location server table and the identification string”). (*Id.*)

75. **Second**, Kahn discloses that the “function [is] commonly known to the first data location server and the first client.” As discussed above and for limitation [10.d], the hash algorithm is commonly known to both the client system (“first client”) and handle servers (including “first data location server”), since both the client system and handle servers encounter the hash algorithm during the digital object retrieval process, e.g., the client generates the hash code for a handle using a hashing algorithm and uses the hash code to index into the hash range/handle-server-domain-name table to determine/calculate the domain name of the handle server. (Section IX.A.1.v; *see also* Sections IX.A.1.i, IX.A.1.iv.) Moreover, the handle servers (including “first data location server”) “know[]” about the hash algorithm because each handle server is assigned a range of handles based on hash code/value ranges, which are the result of the application of the hash algorithm to handles, as shown in Figure 7 discussed for limitation [10.pre]. (*See* Section IX.A.1.i; Ex. 1005, 4:19-22 (“The identifiers are generated in a manner to distributed the pointer information with the unique identifier information[] relatively evenly among the servers, using a hashing algorithm.”), 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.”), 47:1-5 (“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).”); *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.**”).)

4. **Claim 13: “The method of claim 12, wherein the function comprises a hash function and wherein the first client applies the hash function to the identification string and the data location server table to obtain the location of the second data location server.”**

76. Kahn discloses this limitation. As discussed for claim 12, the client system (“first client”) applies a hash algorithm (“hash function”) to the handle (“identification string”) resulting in a hash code/value used to index into the updated hash range/handle-server-domain-name table (“data location server table”) in order to determine the location of the correct handle server (“obtain the location of the second data location server”). (Section IX.A.3.) Kahn’s teaching of applying a hash function to a handle and the updated table is consistent with the ’640 patent’s disclosure, where the hash function is applied to the identifier string and the output thereof is used to index into the server table. (Ex. 1001, 14:64-67 (“The appropriate NDTP server is selected from the table in the NDTP_RDR_RSP message by applying a well-known function to the identifier string and using the function result as an index into the NDTP server table.”).)

5. Claim 17

- i) [17.pre] “A system for retrieving data location information for data stored in a distributed network, the system comprising:”**

77. 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 discloses the limitations therein. As discussed for limitation [10.pre], Kahn discloses a system for retrieving one or more pointers indicating the location of a digital object stored in a repository of a distributed network (“[a] system for retrieving data location information for data stored in a distributed network”). (Section IX.A.1.i.)

- ii) [17.a] “a plurality of data repositories configured to store data, wherein the data is associated with a respective identifier string in each data repository;”**

78. Kahn discloses this limitation. **First**, Kahn discloses “a plurality of data repositories configured to store data.” For example, Kahn discloses multiple repositories (“a plurality of data repositories”) in a network, with each repository storing digital objects (including “data”). (Ex. 1005, 11:16-18 (“A ‘repository’...is a digital storage system into which digital objects...may be placed and retained for possible subsequent retrieval.”); *see also id.*, 6:22-24 (“Owners of...digital objects may deposit copies of the digital objects in the repository, via the network. There

may be multiple repositories.”), 12:19-20 (“A ‘stored digital object’...is a digital object stored in a repository.”); *see* Sections IX.A.1.i-ii.)

79. **Second**, Kahn discloses “wherein the data is associated with a respective identifier string in each data repository.” Kahn discloses that a digital object (including “data”) has an associated transaction record stored in a repository (“data repository”), where the transaction record contains a handle for the digital object (“wherein the data is associated with a respective identifier string”).⁷ (Ex. 1005, 14:33-15:7 (“A stored digital object also may have associated with it, in the repository or elsewhere, an associated ‘transaction record’ 1016 which records transactions involving the digital object. **The transaction record may contain entries such as** the time and date of deposit of the object 1032, the time and date of each request for retrieval of the object, the identity of the requesting party 1034, **the handle 1012** and service request for the object....”).)

⁷ Kahn discloses an “identifier string” under the Interpretation from Other Proceedings for “identifier” for the same reasons as discussed for limitation [10.a]. (*See* Sections VIII, IX.A.1.ii.)

- iii) [17.b] “a data location sewer [sic] network having a plurality of data location servers, each of the plurality of data location servers containing location strings associated with respective identifier strings and each of the plurality of data location servers having computer executable code configured to execute the following steps:”⁸

80. Kahn discloses this limitation. First, Kahn discloses “a data location [server] network having a plurality of data location servers, each of the plurality of data location servers containing location strings associated with respective identifier strings.” As discussed for limitations [10.pre] and [10.a], Kahn discloses a network including multiple handle servers (“data location [server] network having a plurality of data location servers”). (Ex. 1005, 7:5-20, 21:6-24; *see also* above Sections IX.A.1.i-ii.) Kahn also discloses each handle server contains one or more pointers (“each of the plurality of data location servers containing location strings”) associated with a respective handle (“associated with respective identifier strings”). (*See* above Sections IX.A.1.i-iii, IX.A.5.i-ii.)

⁸ I assume that “data location sewer network” is intended to refer to a “data location server network.”

81. **Second**, Kahn discloses “each of the plurality of data location servers having computer executable code configured to execute the following steps” of limitation [17.c]. For example, Kahn discloses each handle server (“each of the plurality of data location servers”) provides handle-to-pointer translation services, i.e., returning a pointer in response to an associated handle, where each handle server can be on one or more computers operating using software. (Ex. 1005, 21:6-15 (“**The overall system...comprises multiple servers which provide...handle-to-pointer translation services (called ‘handle servers’).**”), 26:28-31 (“[A]n example system 31 includes a digital object management system 32 which includes hardware and **software to create and store digital objects and manage rights to the objects.**”), 29:5-12 (“Handle servers 58 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}.”), 42:2-11 (“**[E]ach handle server can be implemented as one or more server computers....**”), 44:6-7 (“A handle is sent to a handle server to locate network addresses of repositories containing that

object.”), 65:14-69:14 (defining various software packages run on the client system).) Moreover, Kahn 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 (“data location server”) provides such services on a computer using “computer executable code” thereof, including the steps recited in limitation [17.c]. (*See* Section IX.A.5.iv.)

- iv) [17.c] “in response to receiving a data location request from a client to retrieve a location string associated with an identification string provided in the data location request, transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server, wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string.”

82. Kahn discloses this limitation. **First**, Kahn discloses “a data location request from a client to retrieve a location string associated with an identification string provided in the data location request.” As discussed for limitation [10.b], a

handle query/request pointer information UDP packet (“data location request”) is transmitted from a client system (“client”) to a handle server to retrieve a pointer (“location string”) associated with a handle (“identification string”)⁹ provided in the handle query/request pointer information UDP packet. (*See* Section IX.A.1.iii ; Ex. 1005, 29:5-12, 47:5-11, 73:32-34; *see also id.*, 7:5-7, 43:20-23, 44:6-7.)

83. **Second**, Kahn discloses “in response to receiving a data location request...transmitting a redirect message to the client if the identification string is not associated with a location string at the data location server.” As discussed for limitation [10.c], Kahn explains that in response to receiving a handle query/request pointer information UDP packet (“data location request”), a not-responsible-for-handle-message (“redirect message” under the Applied Interpretation) is transmitted to the client system (“client”) if the handle (“identification string”) is not associated with a pointer (“location string”) at the handle server (“data location server”), i.e., if the request was sent to the wrong handle server. (*See* Section IX.A.1.iv; Ex. 1005, 47:5-15, 73:21-74:4.)

⁹ I assume that the “identification string” is intended to refer to the “identifier string” recited in earlier limitations of claim 17. (*See* Sections IX.A.5.ii-iii.)

84. **Third**, Kahn discloses “wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string.” As discussed for limitation [10.c], Kahn discloses the not-responsible-for-handle-message (“redirect message” under the Applied Interpretation) containing information for the client system to use (“information for use by the client”) to download an updated table and to determine/calculate the location of a correct handle server containing a pointer locating the digital object (“to calculate a location of a different data location server...wherein the different data location server contains the location string”). (See Section IX.A.1.iv.) As discussed for limitation [17.a], each handle server (including “different data location server”) is one of multiple handle servers (“a plurality of data location servers”). (See Section IX.A.5.ii; *see also* Sections IX.A.1.i-ii.)

6. Claim 18

- i) [18.pre] “A system for retrieving data location information for data stored in a distributed network, the system comprising:”**

85. Kahn discloses this limitation for the same reasons discussed for limitation [17.pre]. (See Section IX.A.5.i.)

- ii) **[18.a] “a data repository configured to store data, wherein the data is associated with an identifier string;”**

86. Kahn discloses this limitation for reasons discussed for limitation

[17.a]. (*See* Section IX.A.5.ii.)

- iii) **[18.b] “a client responsive to a data query to query a data location server for location information associated with the identifier string;”**

87. Kahn discloses this limitation. As discussed for limitations [10.a]-[10.b], Kahn discloses that a client system (“client”) responds to a request (“responsive to a data query”) to retrieve a digital object by sending a handle query/request pointer information UDP packet to a handle server (“to query a data location server”) in order to retrieve one or more pointers (“location

information”)¹⁰ associated with a handle (“associated with the identifier string”).¹¹

(See Sections IX.A.1.ii-iii.)

- iv) **[18.c] “a data location server network comprising a plurality of data location servers, at least one of the plurality of data location servers containing location information associated with the identifier string, wherein each of the plurality of data location servers comprises computer executable code configured to execute the following steps in response to receiving a data location request from the client:”**

88. Kahn discloses this limitation. **First**, Kahn discloses “a data location server network comprising a plurality of data location servers, at least one of the plurality of data location servers containing location information associated with the identifier string.” As discussed for limitation [10.pre], Kahn discloses a network containing multiple handle servers (“a data location server network

¹⁰ Kahn discloses “location information” under the Interpretation from Other Proceedings for the same reasons as discussed for limitation [10.pre]. (See Sections VIII, IX.A.1.i.)

¹¹ Kahn discloses an “identifier string” under the Interpretation from Other Proceedings for “identifier” for the same reasons as discussed for limitation [10.a]. (See Sections VIII, IX.A.1.ii.)

comprising a plurality of data location servers”). (Section IX.A.1.i; Ex. 1005, 7:5-20, 21:6-24.) As further discussed, each handle server (“at least one of the plurality of data location servers”) contains one or more pointers (“containing location information”) associated with a handle (“associated with the identifier string”). For example, Kahn discloses “multiple handle servers, each handle server having a table which associates handles with pointers.” (Section IX.A.1.i; Ex. 1005, 7:17-20; *see also id.*, 21:6-15.)

89. **Second**, Kahn discloses “wherein each of the plurality of data location servers comprises computer executable code configured to execute the following steps” of limitations [18.d] and [18.e]. (*See* below Sections IX.A.6.v-vi.) As discussed for limitations [17.b]-[17.c], each handle server (“each of the plurality of data location servers”) provides handle-to-pointer translation services on a computer using “computer executable code” thereof, including the steps recited in limitation [18.d] and [18.e]. (Sections IX.A.5.iii-iv; *see also* below Sections IX.A.6.v-vi.)

90. As discussed for limitation [10.b] and further below, Kahn discloses that the steps of limitations [18.d] and [18.e] are performed “in response to receiving a data location request from the client.” (*See* Sections IX.A.1.iii,

IX.A.6.v-vi.) Kahn discloses in response to receiving a handle query/request pointer information UDP packet (“data location request”) the client system transmitting a handle to a handle server to retrieve a pointer to the location of a digital object. (Ex. 1005, 29:5-12, 47:5-8; *see also id.*, 7:5-7, 43:20-23, 44:6-7; Section IX.A.6.iii (discussion of handle servers).) Moreover, consistent with as discussed for limitations [10.c]-[10.d], each handle server (“data location server”) performs processes to determine if it contains a particular handle and its associated pointer (“location string”), as further explained below for limitations [18.d] (discussing actions performed if the claimed “data location server” does contain the location string) and [18.e] (discussing actions performed if the claimed “data location server” does not contain the location string). (See Sections IX.A.1.iv-v; below Sections IX.A.6.v-vi.) Accordingly, Kahn discloses this limitation.

- v) **[18.d] “if the data location server contains the location string associated with the identification string provided in the data location request, the data location server transmits location information for use by the client to calculate a location of the data associated with the identification string;”**

91. Kahn discloses this limitation. **First**, Kahn discloses “the data location server contains the location string associated with the identification string provided in the data location request.” For example, Kahn discloses a handle

server (“data location server”) containing one or more pointers (“location string”) associated with a handle (“identification string”)¹² provided in the handle query/request pointer information UDP packet (“data location request”). As discussed for limitations [10.pre] and [10.b], Kahn discloses the client system transmitting a handle in a handle query or request packet to the handle server to determine if the handle server has a pointer associated with the digital object’s handle. (Ex. 1005, 29:5-12, 47:5-8; *see also id.*, 7:5-7, 43:20-23, 44:6-7; Sections IX.A.1.i, IX.A.1.iii.)

92. **Second**, Kahn discloses “if the data location server contains the location string...the data location server transmits location information for use by the client to calculate a location of the data associated with the identification string.” Kahn discloses that if a handle server indeed contains the queried handle and its associated one or more pointers (“if the data location server contains the location string”), the handle server returns the pointers associated with the handle (“the data location server transmits location information”) to the client system for

¹² I assume that “identification string” is intended to refer to the “identifier string” recited in earlier limitations of claim 18. (*See* Sections IX.A.6.ii-iv.)

the client system to determine (“for use by the client to calculate”) the location of the digital object (“a location of the data”) associated with the handle (“identification string”). (See Sections IX.A.1.i-iii.) Kahn discloses “[i]f the handle server is responsible for the handle [and its associated pointer], it verifies that the handle is present in its database.... For each pointer returned by the handle server, the requesting [client] system uses the pointer to attempt to obtain a copy of the digital object.” (Ex. 1005, 74:13-26; *see also id.*, 47:9-11, 73:21-27, FIG. 8 (step 180); Sections IX.A.1.i-iii.) Kahn discloses that the returned pointers include the domain name of the repository that stores the requested digital object. (Ex. 1005, 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, a person of ordinary skill in the art would have understood that to retrieve the digital object, the client system uses the information included in the pointers (“location information for use by the client”), e.g., the repository domain name, in order to determine/calculate the location of the digital object (including “data”) associated with the requested handle (“to calculate a location of the data associated with the identification string”). (See *also id.*, 44:30-32 (“Repository names are not actual network

addresses; they must first be mapped to network locations.”).) Accordingly, Kahn discloses this limitation.

- vi) **[18.e] “if the data location server does not contain the location string associated with the identification string, the location server transmits a redirect message to the client, wherein the redirect message contains redirect information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string.”**

93. **Kahn** discloses this limitation. First, Kahn discloses “wherein the redirect message contains redirect information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data location server contains the location string.” As discussed for limitation [10.c], Kahn discloses that a handle server transmits a not-responsible-for-handle message (“redirect message” under the Applied Interpretation) to the client system (“client”), where the message informs the client to download an updated hash range/handle-server-domain-name table (“wherein the redirect message contains redirect information for use by the client”), for use by the client system in determining/calculating a correct handle server containing a pointer to the location of the digital object (“to calculate a location of a different

data location server in the plurality of data location servers, wherein the different data location server contains the location string”). (See above Section IX.A.1.iv.)

94. **Second**, Kahn discloses “if the data location server does not contain the location string associated with the identification string, the location server transmits a redirect message to the client.” As discussed for limitation [10.c], Kahn discloses that if a handle server (“data location server”) does not contain the handle (“identification string”) associated with one or more pointers (“location string”), the handle server transmits a not-responsible-for-handle message (“redirect message” under the Applied Interpretation) to the client system (“client”). (See Section IX.A.1.iv; Ex. 1005, FIG. 8 (step 182), 47:13-15, 73:32-74:4.)

7. **Claim 19: “The system of claim 18, wherein the client and the plurality of data location servers each comprise a function commonly known to the client and the plurality of data location servers and wherein the client is configured to apply the commonly known function to the location information or redirect information.”**

95. Kahn discloses this limitation. As discussed for limitation [10.d], claim 12, and claim 13, Kahn discloses a hash algorithm (“function”) commonly known to the client system (“client”) and multiple handle servers (“plurality of data location servers”), as the client system and handle servers either apply or are

associated with the hash algorithm while performing the digital object retrieval process (“wherein the client and the plurality of data location servers each comprise a function commonly known to the client and the plurality of data location servers”). (See Sections IX.A.1.v, IX.A.3, IX.A.4.) As further discussed therein, the client system (“client”) applies the hash algorithm to a handle, resulting in a hash code/value used to index into the hash range/handle-server-domain-name table (“redirect information”) to determine the domain name of the correct handle server containing a handle and its associated pointer to the location a digital object (“wherein the client is configured to apply the commonly known function to the location information or redirect information”). (See Sections IX.A.1.v, IX.A.3, IX.A.4.) Accordingly, Kahn discloses this limitation.

8. Claim 24: “The system of claim 18, wherein the location information comprises a portion of a hash table distributed over the plurality of data location servers.”

96. Kahn discloses this limitation. To begin, Kahn discloses that all handles are partitioned amongst multiple handle servers, where the handles are partitioned/distributed based on the hash algorithm and hash range/handle-server-domain-name table, such that each handle server is assigned a portion of the handle space. (Ex. 1005, 7:5-9 (“A handle server, accessible on the network, provides the

pointer in response to presentation of a handle. **Multiple servers provide the service, each serving a portion of the handle space.**”), 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.”), 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 (“The system...generates the hash code for the handle using a predetermined hashing algorithm...and consults the hash range/handle-server-domain-name table to determine the domain name of the appropriate handle server....”).) Thus, handles and their associated pointers (“location information”), which are stored in handle/pointer mappings on the handle servers, are likewise partitioned/distributed over multiple handle servers according to the hash range/handle-server-domain-name table, such that the pointers on a given handle server represent only a portion of the full set of pointers. A person of ordinary skill in the art would have therefore understood that the pointers (“location

information”) on a given handle server comprise a portion of the full handle-server-domain-name table (“hash table”) distributed over the plurality of handle servers (“comprises a portion of a hash table distributed over the plurality of data location servers”) because the pointers are associated with a portion of the handles in the table.

B. Kahn in View of Vingralek Discloses and/or Suggests Claims 10-24

1. Claim 10

97. Kahn in view of Vingralek discloses and/or suggests claim 10. As discussed in Section IX.A.1, Kahn discloses claim 10. (*See* Section IX.A.1.) I have been asked to consider a scenario where limitation [10.c] is interpreted to require that the “redirect message” is sent from a “data location server” and/or Kahn’s “redirect message” does not disclose the redirect message “containing information for use by the first client to calculate a location of a second data location server.” In that scenario, a person of ordinary skill in the art would have been motivated to implement these features in view of Vingralek.

98. 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.

99. Vingralek additionally discloses propagating updates to clients’ and other servers’ address tables and hash functions in a deferred manner. (Ex. 1007, 6.) To limit communication overhead, clients learn about changes to the hash function and associated 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 hash function and 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 hash function and address table are deferred until a wrong first server is reached and the location of a second server needs to be calculated.

100. Vingralek’s method of deferred hash function and 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).”.) Similarly, each client and server may have stale information on the hash function. (*Id.*, 5 (“client’s stale information on the hash function”), 9 (describing the “file level” as a parameter derived from the index to the hash function), 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 file level...).”.) When a client seeks to retrieve records

associated with a given key, it applies its current 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 its 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 hash function and address table. (*Id.*; *see also id.*, 25 (referring to "client requests and server responses" as "messages").)

101. 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 and the parameters for the hash function are recalculated. (*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 and hash function each time a request is forwarded to the next server (until the request reaches a correct server containing the requested information, i.e., the "second data location server") would have gradually brought the client's perception of the address table and hash function 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 hash function and address/hash table, which the client uses when calculating the location of a server.

102. Additionally, Vingralek discloses that "[t]he most salient feature of [the LH*] approach," where a client's hash function and table are 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.)

103. 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 function and 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 and hash function are 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 and hash function update, as it would have allowed the client to use a correct/updated hash table and hash function 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, 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.”).) Thus, a person of ordinary skill in the art would have been motivated to include deferred hash table and hash function 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 and function (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.”.)**)

104. 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 limitation [10.c], 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.1.iv.) Combining the transmission of the message from the handle server with the update to the client’s hash function/table would have reduced the number of required interactions between system components, thereby reducing communication overhead.

105. Additionally, a person of ordinary skill in the art would have been motivated to implement the deferred hash function/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 (“first client”) would apply its current hash function to its downloaded copy of the hash table, and send a request

to the first handle server (“first data location 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 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 the handle server has the requested location information. If the first handle server does not have the requested location information, i.e., the pointers associated with the handle (“if the first data location server does not possess the location string”), the handle server would transmit a message (“redirect message”) to the client system, including the handle server’s hash table and hash function for use by the client to calculate a location of a second handle server that contains the pointers associated with the handle (“transmitting a redirect message to the first client, the redirect message containing information for use by the first

client to calculate a location of a second data location server, wherein the second data location server contains the location string”). The client system would then calculate the location of the correct handle server (“calculating the location of the second data location server at the first client”) and send the request to that correct handle server (“transmitting the data query from the first client to the second data location server”), as discussed in Sections IX.A.1.v-vi, limitations [10.d]-[10.e]. (Sections IX.A.1.v-vi.) 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 a redirect message with information *for the client* (“first 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/function updates to propagate from handle server to client, while reducing the number of updates, e.g., downloads performed from the handle directory servers (i.e., by reducing the number of downloads by the client system).

106. 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 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 functions and 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 11

107. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 10 in Section IX.B.1, the Kahn-Vingralek

combination discloses and/or suggests a first handle server that does not have the requested location information would send a message (“transmitting the redirect message”) to the client system (“first client”) in order to update the client’s hash function and hash table. (*See* Section IX.B.1.) A person of ordinary skill in the art would have understood that, to update the hash table, the redirect message could include either the specific information to be updated or the entire table for replacing the table depending on, e.g., the extent of the update required. Thus, a person of ordinary skill in the art would have been motivated to implement transmitting an entire hash range/handle-server-domain-name table (a hash table or “data location server table”) to the client (“first client”). Indeed, Vingralek suggests that the entire hash table may be sent to the client in order to update a stale table. (Ex. 1007, 11 (“However, if [key] *K* has been sent to the wrong server, the server...updates the client’s view with...its address table.”).) Thus, the Kahn-Vingralek combination discloses and/or suggests “transmitting the redirect message comprises transmitting a data location server table to the first client.”

3. Claim 12

108. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 11 in Section IX.B.2, a person of ordinary skill

in the art would have been motivated to implement the redirect message such that it contains a hash table (“data location server table”) that the client (“first client”) uses to calculate the location of a correct handle server (“calculating the location of the second data location server”) using a hash function. (*See* Section IX.B.2.) As discussed for claim 10 in Section IX.B.2, both client and handle servers (including “first data location server”) apply the function in the retrieval process (the client to determine where to send a request and the handle server to determine if it has pointers associated with a handle), and updates to the function (e.g., because of changes in the buckets or servers) are shared among the client and handle servers so each component has an up-to-date perception of the function (“a function commonly known to the first data location server and the first client”). (Section IX.B.1.) Thus, the hash function is commonly known to both the client (“first client”) and the handle servers (including “first data location server”). As further discussed therein, the updated hash function is applied by the client to the newly updated hash table and the handle (“based on the data location server table and the identification string”) in order to determine the location of the second handle server (“calculating the location of the second data location server”). (*See id.*)

4. Claim 13

109. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 12 in Section IX.B.3, the disclosed “function” comprises a hash function (“wherein the function comprises a hash function”). (Section IX.B.3.) Moreover, as discussed therein, the client system (“first client”) applies the updated hash function to the handle (“identification string”) and updated hash table (“data location server table”) in order to determine the location of the second handle server (“obtain the location of the second data location server”). (*See id.*; *see also* Section IX.A.4.)

5. Claim 14: “The method of claim 10, wherein transmitting the redirect message comprises transmitting a function to the first client.”

110. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 10 in Section IX.B.1, the Kahn-Vingralek combination discloses and/or suggests sending a message (“transmitting the redirect message”) that includes an updated hash function (“transmitting a function”) and hash table to the client system (“first client”) in order to update the client’s hash function (“the redirect message comprises transmitting a function to the first client”) and hash table. (*See* Section IX.B.1.)

6. Claim 15: “The method of claim 14, wherein calculating at the first client the location of the second data location server comprises applying the transmitted function to the identifier string.”

111. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 10 in Section IX.B.1, the client system calculates a location of a correct handle server (“calculating at the first client the location of the second data location server”) by applying the transmitted/updated hash function (“transmitted function”) to the handle (“identifier string”).¹³ (See Section IX.B.1.)

7. Claim 16: “The method of claim 15, wherein applying the transmitted function generates a URL of the second data location server.”

112. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for limitation [10.d] in Section IX.A.1.v, Kahn discloses applying the hash algorithm to the handle for indexing into the hash range/handle-server-domain-name table to generate the domain name/network address of the correct handle server (“URL of the second data location server”) containing the

¹³ I assume the “identifier string” is intended to refer to the “identification string” recited claim 10. (Sections IX.A.1.ii-iii.)

handle. (See Section IX.A.1.v; *see also* Ex. 1005, 38:11-14 (“The directory servers will maintain a table of network addresses of handle servers (generally, each handle server will contain such a directory).”), 46:22-47:5 (“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....**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).**”).) A person of ordinary skill in the art would have understood that the process of applying the transmitted/updated hash function (“transmitted function”) to the handle, as discussed for claim 15, would have likewise generated the domain name/network address of the correct handle server (“URL of the second data location server”) containing the handle. (Section IX.B.6.) Moreover, Kahn discloses that its system, which includes a handle server (“data location server”), may be used on a network such as the Internet. (Ex. 1005,

2:25-27 (“Making digital objects available on **networks (e.g., Internet)...**”),
19:31-20:2 (“[T]he network medium could be organized as a single local area
network, a wide area network, or a **broader network structure (e.g., the
Internet).**”); *see also id.*, 7:5-7 (“A **handle server, accessible on the network,**
provides the pointer in response to presentation of a handle.”).) A person of
ordinary skill in the art would have understood that a network address on the
Internet is a URL, and thus the location of any server connected by the Internet
(e.g., the handle servers) is defined by such a URL. Accordingly, the Kahn-
Vingralek combination discloses and/or suggests this limitation.

8. Claim 17

113. The Kahn-Vingralek combination discloses and/or suggests this
limitation. As discussed in Section IX.A.5, Kahn discloses claim 17. (*See* Section
IX.A.5.) I have been asked to consider a scenario where Kahn’s redirect message
described for limitation [17.c] does not disclose that the redirect message contains
information for the client for use to calculate the location of a different location
server. 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 10 in Section IX.B.1. (Section IX.B.1.)

114. For example, as discussed for claim 10 in Section IX.B.1, the Kahn-Vingralek combination discloses and/or suggests that, after the first handle server receives the request from the client (“in response to receiving a data location request from a client”) to retrieve pointers associated with a handle provided in the request (“to retrieve a location string associated with an identification string provided in the data location 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 (“location string”) associated with the handle (“identification string”). (See Section IX.B.1.) As further discussed, if the first handle server does not have the requested pointers associated with the handle (“if the identification string is not associated with a location string at the data location server”), the handle server would send a message to the client system (“transmitting a redirect message”) to update the client’s hash function and hash table, thus allowing the client to use the updated hash function to calculate the correct handle server for sending the request (“wherein the redirect message contains information for use by the client to calculate a location of a different data location server in the plurality of data location servers, wherein the different data

location server contains the location string”). (*See id.*) Accordingly, the Kahn-Vingralek combination discloses and/or suggests this limitation.

9. Claim 18

115. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed in Section IX.A.6, Kahn discloses claim 18. (*See* Section IX.A.6.) I have been asked to consider a scenario where Kahn’s redirect message described for limitation [18.e] does not disclose that “the redirect message contains redirect information for use by the client to calculate a location of a different data location server.” In that scenario, a person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek. (Section IX.B.1.)

116. As discussed for claim 10 in Section IX.B.1, the Kahn-Vingralek combination discloses that, after the first handle server receives the request, the handle server applies the hash function to the handle and to its copy of the hash table to determine whether it has the requested pointers (“location string”). (*See* Section IX.B.1.) As further discussed, if the first handle server does not have the requested pointers associated with the handle (“if the data location server does not contain the location string associated with the identification string”), the handle server would send a message to the client system (“the location server transmits a

redirect message to the client”) containing information, i.e., updated hash function and hash table (“redirect information”) for updating the client’s hash function and hash table. (*See id.*) This step would allow the client system to use the updated hash function and hash table to calculate the location of the correct handle server in the system of multiple handle servers (“to calculate a location of a different data location server in the plurality of data location servers”) to send the request to, where the “correct” handle server contains the requested pointers (“wherein the different data location server contains the location string”). (*See* Section IX.B.1.) Accordingly, the Kahn-Vingralek combination discloses and/or suggests this limitation.

10. Claim 19

117. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 12 in Section IX.B.3, the Kahn-Vingralek combination discloses and/or suggests that a hash function is known to both the client and the handle servers, since both the client and handle servers apply the function in the retrieval process and updates to the function are shared among client and handle servers (“wherein the client and the plurality of data location servers each comprise a function commonly known to the client and the plurality

of data location servers”). (*See* Section IX.B.3.) As further discussed for claim 11 in Section IX.B.2, a person of ordinary skill in the art would have been motivated to implement the redirect message such that it contains a hash table and hash function (“redirect information”) that the client system (“client”) uses to determine a correct handle server by applying the hash function to the handle and hash table (“wherein the client is configured to apply the commonly known function to the...redirect information”). (*See* Section IX.B.2.)

11. Claim 20: “The system of claim 19, wherein the redirect message comprises a data location server table.”

118. The Kahn-Vingralek combination discloses and/or suggests this limitation for the reasons given in Section IX.B.2, claim 11. (*See* Section IX.B.2.)

12. Claim 21: “The system of claim 20, wherein the commonly known function comprises a hash function and wherein the client is configured to apply the hash function to the identification string and the data location server table to obtain the location of the different data location server.”

119. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 19 in Section IX.B.10, the hash function known to the client system and handle servers (“wherein the commonly known function comprises a hash function”) is applied by the client system (“client”) to the handle (“identification string”) and hash table (“data location server table”) to determine

the location of a second handle server (“to obtain the location of the different data location server”). (*See* Section IX.B.10.)

13. Claim 22: “The system of claim 19, wherein the redirect message comprises a transmitted function for use by the client.”

120. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claims 14 and 15, the Kahn-Vingralek combination discloses and/or suggests the handle server transmitting a message (“redirect message”) that includes an updated hash function (“comprises a transmitted function”) for use by the client system (“for use by the client”) in calculating the location of the correct handle server. (*See* Sections IX.B.5, IX.B.6.)

14. Claim 23: “The system of claim 22, wherein the client is configured to calculate the location of the different data location server by applying the transmitted function to the identifier string.”

121. The Kahn-Vingralek combination discloses and/or suggests this limitation. As discussed for claim 15, the Kahn-Vingralek combination discloses and/or suggests the client system calculating the location of another handle server (“wherein the client is configured to calculate the location of the different data location server”) by applying the hash function transmitted in the redirect message

to the handle (“by applying the transmitted function to the identifier string”). (*See* Section IX.B.6.)

15. Claim 24

122. Kahn discloses this limitation for the reasons given in Section IX.A.8, claim 24. (*See* Section IX.A.8.)

X. CONCLUSION

123. 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