

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

GOOGLE LLC,
Petitioner

v.

KOVE IO, INC.,
Patent Owner

Patent No. 7,233,978

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

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	BACKGROUND AND QUALIFICATIONS	1
III.	MATERIALS REVIEWED	6
IV.	PERSON OF ORDINARY SKILL IN THE ART AND THE TIME OF THE ALLEGED INVENTION	8
V.	TECHNICAL BACKGROUND	9
	A. Distributed Computing Systems	10
	B. Hash Functions and Applications	12
VI.	THE '978 PATENT	15
	A. Description	15
	B. The '896 Application Does Not Support All the Features of Claims 10-17, 23-26, and 30	18
VII.	OVERVIEW OF THE PRIOR ART	21
	A. Skagerwall	21
	B. Vingralek	27
	C. Krasner	30
	D. Sato	33
VIII.	CLAIM CONSTRUCTION	35
	A. "identifier"	37
	B. "location information"	37
	C. "location server"	37
	D. "client"	37
	E. "plurality of location servers"	38
IX.	THE PRIOR ART DISCLOSES AND/OR SUGGESTS ALL RECITED FEATURES OF CLAIMS 10-17, 23-26, and 30 OF THE '978 PATENT	40

A.	Skagerwall in View of Vingralek Discloses and/or Suggests Claims 10, 12-17, 23, and 30	40
1.	Claim 10	40
2.	Claim 12	77
3.	Claim 13	77
4.	Claim 14	78
5.	Claim 15	84
6.	Claim 16	84
7.	Claim 17	86
8.	Claim 23	101
9.	Claim 30	102
B.	Skagerwall in View of Vingralek and Krasner Discloses and/or Suggests Claim 11	103
1.	Claim 11	103
C.	Skagerwall in View of Vingralek and Sato Discloses and/or Suggests Claims 24-26	108
1.	Claim 24	108
2.	Claim 25	112
3.	Claim 26	113
X.	CONCLUSION.....	116

I, Sandeep Chatterjee, declare as follows:

I. INTRODUCTION

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

II. BACKGROUND AND QUALIFICATIONS

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

III. MATERIALS REVIEWED

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

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

Ex. 1001	U.S. Patent No. 7,233,978 to Overton et al. (“the ’978 patent”)
Ex. 1003	Curriculum Vitae of Sandeep Chatterjee, Ph.D.
Ex. 1004	Prosecution History of U.S. Patent No. 7,233,978
Ex. 1005	WIPO Application No. 1997/043717 to Kahn et al.
Ex. 1006	U.S. Patent No. 6,473,781 to Skagerwall et al.
Ex. 1007	R. Vingralek, <i>Distributed file organization with scalable cost/performance</i> (1993)
Ex. 1009	U.S. Patent No. 6,104,338 to Krasner
Ex. 1011	Japanese Patent Application Publication No. H09-231180 to Sato et al.
Ex. 1021	U.S. Patent Application No. 09/111,896
Ex. 1056	Joseph B. Greene, <i>Oracle8 Server Unleashed</i> (1998)
Ex. 1057	Joseph B. Greene, <i>Oracle DBA Survival Guide</i> (1995)
Ex. 1058	Betty Salzberg, <i>File Structures</i> (1988)
Ex. 1059	Andrew P. Black & Yeshayahu Artsy, <i>Implementing Location Independent Invocation</i> (1989)

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

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

specification of the '978 patent at the time of the alleged invention, which I discuss below.

18. I understand that the earliest application associated with the '978 patent is U.S. Patent Application No. 09/111,896 ("the '896 application"), which was filed on July 8, 1998. (Ex. 1001, 1:9-26.) As discussed in Section VI.B below, in my opinion, a person of ordinary skill in the art reviewing the '896 application would *not* have understood that the named inventor was in possession of the subject matter recited in claims 10-17, 23-26, and 30 of the '978 patent. Thus, I have been asked to consider that the time of the alleged invention of the '978 patent was around 1999 (including and up to August 13, 1999), which corresponds to the filing date of Provisional Application No. 09/367,461 for the '978 patent. (*Id.*)

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

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

20. I am familiar with the level of ordinary skill in the art regarding the '978 patent as of around August 13, 1999. Considering the '978 patent, the technology, the educational level and experience of workers in the field relating to

the patent, and problems and solutions in that field, and drawing on my own experience, I believe a person of ordinary skill in the art at the time of the alleged invention (around August 13, 1999) would have had an undergraduate degree in electrical engineering, computer engineering, computer science or a related field along with two years of work experience in the field of distributed computing systems.

21. My opinions in this Declaration regarding the '978 patent and the prior art (including the state of the art) are from the perspective of one of ordinary skill in the art as I defined above, during the relevant timeframe (e.g., the time of the alleged invention), which I discussed above as being around August 13, 1999.

V. TECHNICAL BACKGROUND

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

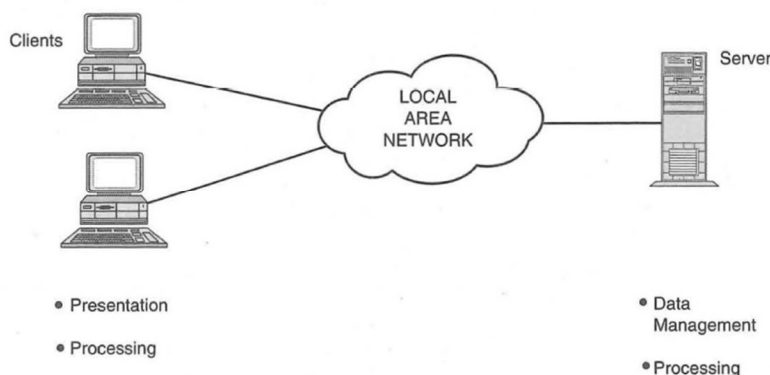
those opinions relating to how the prior art discloses and/or suggests the challenged claims of the '978 and how and why a person of ordinary skill in the art would have been motivated to consider and combine the disclosures and suggestions from that prior art as I explain below in Section IX. The concepts below would have been within the knowledge and mindset of a person of ordinary skill in the art at the time.

A. Distributed Computing Systems

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

at many sites, it is still one database at the highest level that is simply distributed physically...."); Ex. 1001, 12:46-49 (disclosing that hash functions were well known at the time of the alleged invention.) “In distributed databases, you have multiple, completely equal databases,” where the distributed databases may use a client-server-architecture to accomplish tasks. (Ex. 1057, 680.) As shown with reference to Figure 46.1 below, a client-server architecture in its most simple form includes “two intelligent devices (your desktop PC and the server) that share the work load between them.” (Ex. 1057, 674.)

*Figure 46.1.
Client-server
architecture.*



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

24. Distributed network application servers can take applications and provide “reliable, scalable, and robust applications services.” (Ex. 1056, 36 (“Network computing is about universal access to data...universal access to

applications...network services...and bringing new applications online faster and at a lower cost.... Powerful applications servers will take simple objects such as Enterprise JavaBeans and turn them into reliable, scalable, and robust applications services.”.) In particular within a distributed network, “load distribution and load balancing are basic means of improving performance.” (Ex. 1011, ¶[0003] (“Furthermore, load distribution and load balancing are basic means of improving performance in distributed systems.”).)

B. Hash Functions and Applications

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

26. “The point of hash functions is to take a highly nonuniform distribution—probably one with data unevenly clustered—and transform it into one which is nearly uniform.” (*Id.*, 203.) For example, if data is stored non-linearly, a hash function is designed to be “fast for fetching, inserting, and deleting” files within a system. (*Id.*, 215.) Hash functions are also optimal “[w]hen a large number of individual records must be fetched.” (*Id.*, 241 (“Hashing is the best file structure for single-record fetches. **When a large number of individual records must be fetched**, as with automatic teller machines or airline reservations, **this is the best method.**”); *see also id.*, 215 (“Usually, right after a hash table has been created, fetching takes about one disk access. This is the method of choice for many files and many database management systems. **When most of the operations on the file are individual-record fetches, hashing is the optimal file structure.**”).)

27. Hash functions are used in many applications, but as an example, a hash function can be used to locate a tax record associated with a social security number (SSN) within a distributed system, where the SSNs are not stored numerically. (*Id.*, 209 (“For example, suppose that the California tax records are organized by a hash table based on social security number. This means that there is a randomizing function calculated from the social security number and a remainder

taken from this number. The records are *not* stored in order of social security number. Two records whose social security numbers differ only in the last digit may be nowhere near each other.”.) To locate a tax record associated with a SSN, a hash function is applied to the SSN. (*Id.*) The result of the hash function is then used to fetch the tax record. (*Id.*, 209-210 (“If we want to read the records of all the California taxpayers whose social security number is between 000-00-0000 and 000-99-9999 from the file of California tax records, we...have to make 1 million separate fetches for the 1 million different numbers between 0 and 999,999..... Making one million hash-table fetches will take about 7 hours.”), 242 (“For creating a hash table from an already existing large file, as happens initially for the linear-hashing table or repeatedly for the traditional hashing method, we need to sort the records by their hash value and create the table in the last pass.”).)

28. When data is not found at the specified location, perhaps due to the data being stored in a new location in the system, a message with a forwarding address may be returned to the requesting system. (Ex. 1059, 553-54 (“If object *o* is at node *n* after its a^{th} move, then the pair $\langle n, a \rangle$ is a *temporal address descriptor* (*tad*) for *o*.... If *o* later migrates to *m*, its descriptor on *n* is updated to include new *tad* $\langle m, 1 \rangle$. Assuming that objects do not migrate too often, this *tad* represents a forwarding address that is likely to remain correct for some time.... **If *o* is no**

longer at m , then m will normally have a newer tad for o . What should m do?

There are two alternatives: (1) return a failure indication along with its stashed tad , or (2) propagate the invocation to the location indicated in its tad .

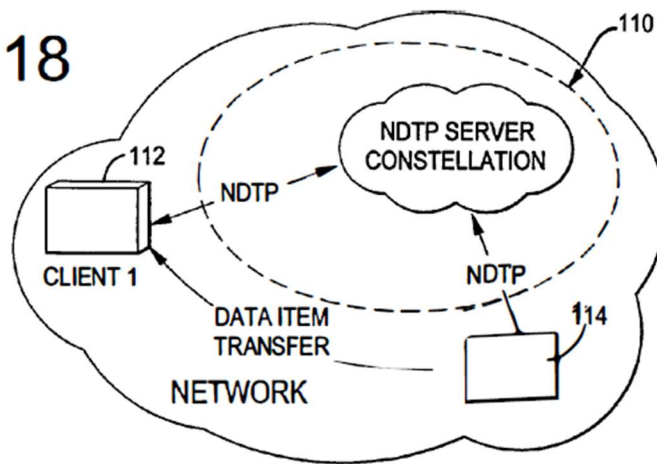
In Hermes, we chose a hybrid of the two approaches. If the invoked node does not have the object, the invocation is propagated to the location indicated in the tad ; this continues up to some small number of hops.... **Otherwise, a failure with a newer tad is returned to the originator of the search, n in this example, which starts afresh with the new forwarding pointer.”.)** For example, if the tax record is no longer stored at the original location, a message can be returned to the requesting system with the updated location. (*Id.*)

VI. THE '978 PATENT

A. Description

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

FIG. 18

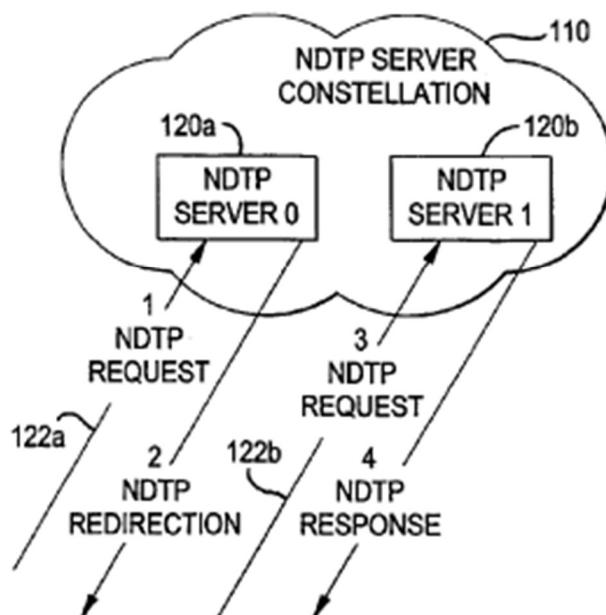


(*Id.*, FIG. 18.)

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

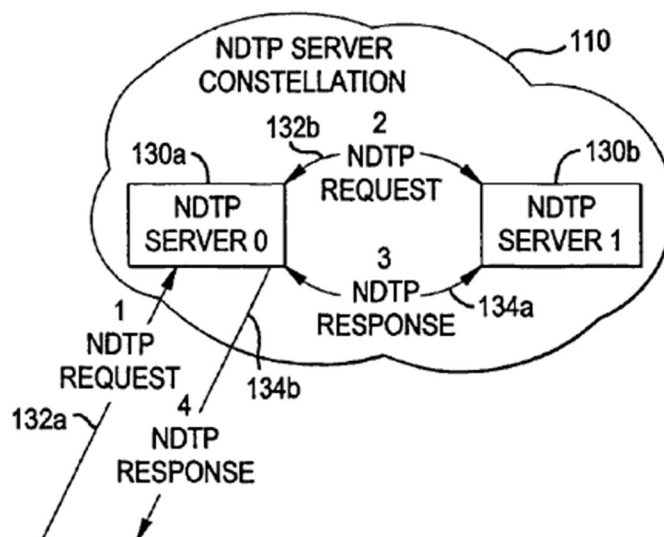
(*Id.*, 18:60-19:3.)

FIG. 19



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

FIG. 20



(*Id.*, FIG. 20.)

31. In both the client-centric and server-centric approaches, a client transmits an identifier specifying an entity to a server in a client request. (Ex. 1001, 2:19-52.) If the server contains location information specifying network locations of data associated with the entity, the server returns the location information. (*Id.*)

B. The '896 Application Does Not Support All the Features of Claims 10-17, 23-26, and 30

32. I understand that the '978 patent issued from U.S. Patent Application No. 09/872,736, filed on June 1, 2001, which claims priority to Provisional Application No. 60/277,408 filed on March 19, 2001; Application No. 09/661,222

(now the '640 patent) filed on September 13, 2000; Provisional Application No. 60/209,070 filed on June 2, 2000; Application No. 09/503,441 filed on February 14, 2000; Application No. 09/367,461 filed on August 13, 1999; and the '896 application filed on July 8, 1998. (Ex. 1001, Cover, 1:9-26.)

33. I have been asked to consider whether a person of ordinary skill in the art reviewing the '896 application would have understood that the named inventors were in possession of the subject matter recited in claims 10-17, 23-26, and 30 of the '978 patent at the time of the '896 application, i.e., July 8, 1998. As discussed below, it is my opinion that the '896 application does not support the features of the challenged independent claims of the '978 patent, and thus does not support the features of any of the challenged claims of the '978 patent.

34. In particular, it is my opinion that none of the challenged independent claims of the '978 patent (i.e., claims 10, 14, and 17) are disclosed in the '896 application; therefore, none of the challenged claims of the '978 patent are entitled to claim priority to the '896 application.

35. The '896 application does not provide written description support for at least the following limitations of the '978 patent: limitations [10.b], [14.c], [17.pre], and [17.e]. Limitations [10.b] and [14.c] require that if a location server does not contain the requested location information associated with the desired

identifier or entity in the request, the server returns a redirect message with information for finding a location server that does contain the requested location information. The '896 application does not provide support for any such redirect message, nor does it provide support for any of the operations associated with a redirect message. At most, the '896 application describes a "server response message[]" that "answers client queries, and sometimes also performs actions on behalf of the client called server actions." (Ex. 1021, 15:23-26.) Although the '896 application discloses that server response messages may be sent in response to a number of events, it does not disclose the claimed "redirect message" and associated operations recited in claims 10 and 14, as described above. (*Id.*, 15:27-24:24, 29:7-41:10.) Limitations [17.pre] and [17.e] require a method of scaling capacity or transaction rate capability in a location server, where a portion of identifiers and associated locations are transferred from a first to second location server when a performance criterion of the first location server is reached. The '896 application does not provide support for any method of scaling capacity or transaction rate capability in a location server, nor providing support for transferring identifiers and locations to another location server when a performance criterion is reached. At most, the '896 application describes that the system is scalable. (*Id.*, 10:3-6, 11:21-12:3, 13:13-15.)

36. Accordingly, it is my opinion that a person of ordinary skill in the art reviewing the '896 application would *not* have understood that the named inventors were in possession of the subject matter recited in challenged claims 10-17, 23-26, and 30 of the '978 patent. Based on my opinion on this matter, I have been asked to consider that the time of the alleged invention of the '640 patent was around 1999 (including and up to August, 13, 1999), which corresponds to the filing date of Application No. 09/367,461 filed on August 13, 1999 for the '978 patent.

VII. OVERVIEW OF THE PRIOR ART

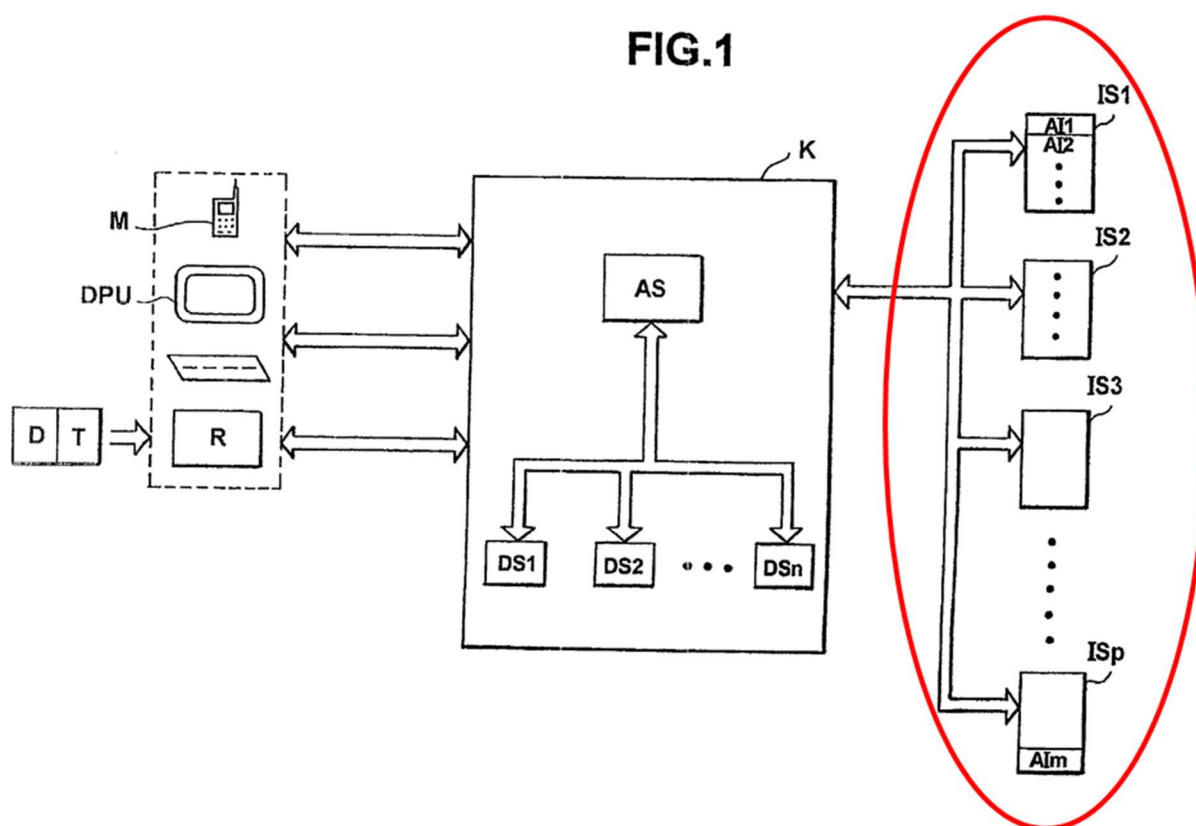
A. Skagerwall

37. Skagerwall discloses a communication system that links objects, such as products or items in the physical world, to different types of information related to the objects. (Ex. 1006, Abstract (“[c]ommunication system and method for linking each of a plurality of tagged object[s] to information services stored on a network”), 6:45-48 (“[t]he system is arranged to link the plurality of objects, which can, for example, be any products or items of the physical world to various kinds of information related to the objects”).) The system allows for linking an arbitrary number of objects to information or application data packets about those objects, where the application data packets are stored on application servers. (*Id.*, 4:1-5

(“The invention advantageously allows to link an arbitrary number of objects of the real world to information services stored on an arbitrary number of data processing units or application servers, which may be located at arbitrary places.”), 4:6-11 (“The data processing units or application servers preferably store a plurality of application data packets comprising data related to the services. Each of the plurality of application data packets may be associated with one of the plurality of objects of the real world, i.e., it contains information related to the object.”).) The application data packets may include text, images, video, audio, application programs, or other kinds of messages. (*Id.*, 4:5-9 (“The services may be text, images, application programs or any other kind of messages.”), 7:13-17 (“Data stored on the application servers can, for example, be text, such as a user manual of a product, links to other servers storing related data or other kind of written information. Further, the data can comprise images, video, audio signals and the like.”).)

38. Skagerwall discloses that the application data packets are stored in a distributed network of application servers. For example, Skagerwall discloses with respect to Figure 1 that the application data packets, each identified by an application identifier (shown as AI1-AIm below), are distributed across a plurality of application servers, each identified by an address (shown as IS1-ISp below).

(*Id.*, 6:34-37 (“The system kernel K is arranged to communicate data between a plurality of application servers and a telephone M, a data processing unit DPU or the like. The application servers each have an address IS; IS1-ISp and store a plurality of application data packets, each having an application identifier AI1-Aim.”).)



(*Id.*, FIG. 1 (annotated to show application servers, identified by addresses IS1-ISp, and the application data packets, identified by identifiers AI1-Aim, in red).)

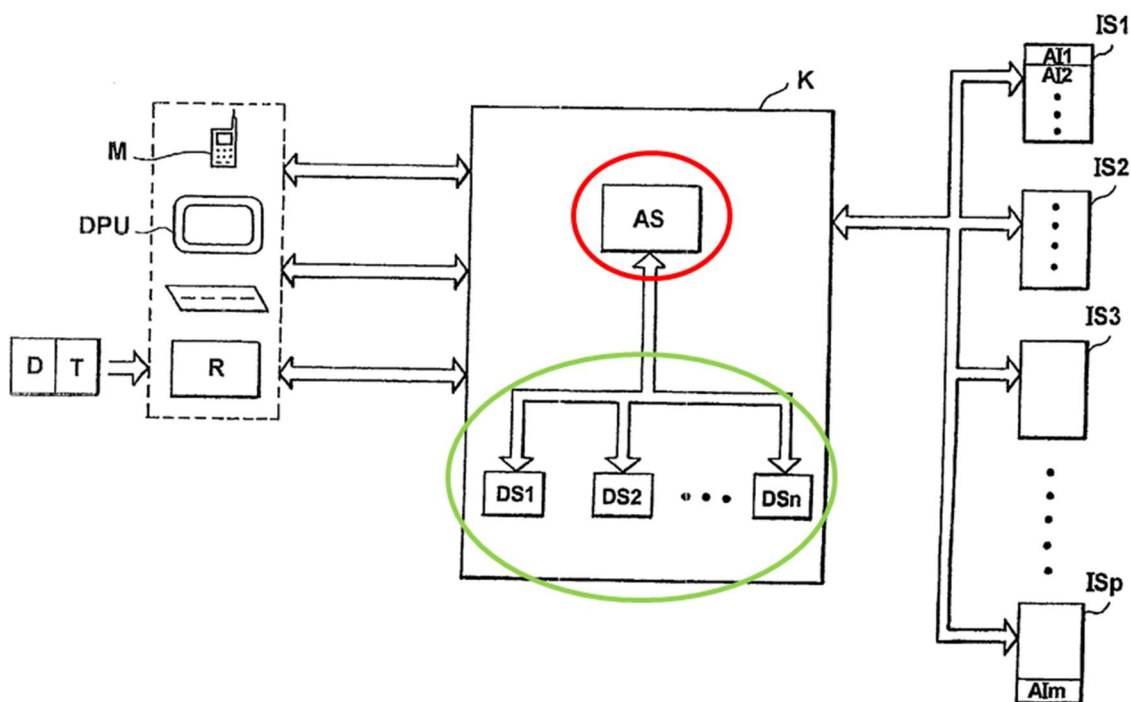
The application servers are arranged in a distributed network because computing resources and data are spread across multiple servers, with servers able to

communicate with one another without going through a centralized point. Indeed, Skagerwall discloses that the application servers may be placed at arbitrary locations, similar to computer networks such as intranets, company networks, or the Internet. (*Id.*, 7:4-8 (“The plurality of application servers with the addresses IS; IS1-ISp and storing data related to the objects D may be placed at arbitrary locations in the world, similar to today’s computer networks, such as intranets, company networks, or the Internet.”).)

39. Skagerwall discloses with respect to Figure 1 that the system includes an access server and a network of directory servers. (*Id.*, 6:32-34 (“The embodiment of FIG. 1 comprises a system kernel K with at least one [access] server AS; AS1-ASr and a plurality of directory servers DS; DS1-DSn.”).)²

² A person of ordinary skill in the art would have understood that the reference to “server AS; AS1-ASr” as “application server[s]” is a typographical error, as these are referred to as “access server[s]” elsewhere in the specification, while the servers IS; IS1-ISp are referred to as the addresses of “application servers.” (*See, e.g., Ex. 1006, 9:1-9.*)

FIG.1



(*Id.*, FIG. 1 (annotated to show access server AS in red and directory servers DS1-DSn in green).) Each directory server stores object data, which associates objects with application data packets, and serves as a key resource in the retrieval of application data packets from application servers. (*Id.*, 4:13-16 (“[E]ach of a plurality of directory servers stores part of an arbitrarily scaleable data base containing object data associating each of the objects with at least one of the application data packets. The plurality of directory servers serve as [a] key resource in retrieving an application data packet related to an object.”), 8:19-22 (“Each of the directory servers DS; DS1-DSn preferably stores object data related

to a small number of the range of available object tags T; T1-Tw.”), FIG. 1; *see also id.*, Abstract (“The system reads an object tag of one of the objects and retrieves information about the object stored on storage located at arbitrary places.”).) Skagerwall explains that the object data is associated with an object tag and stored on a directory server, and the object data includes application identifiers of application data packets and addresses of the application servers storing those packets, which together identify where an application data packet is stored. (*Id.*, 8:22-27 (“[O]bject data related to an object tag preferably includes application identifiers A11-A1m of application data packets related to the object as well as information about storage locations of the application data packets, e.g., the addresses of application servers IS; IS1-ISp storing the application data packets.”), 15:15-52 (explaining how the location of a given application data packet on a network of application servers is characterized by the application server address as well as the application identifier).)

40. Skagerwall discloses that these application identifiers and addresses of application servers identifying application data packets stored in application servers may be retrieved as one part of a two-step process for the retrieval of application data packets. (*Id.*, 8:28-36.) The first step comprises identifying a directory server storing object data (which includes application identifiers and

addresses) relating to an object, and retrieving that object data. (*Id.*, 8:22-32; *see also id.*, 9:5-9 (“[T]he at least one access server AS; AS1-ASr is responsible for retrieving addresses IS; IS1-ISp of application servers obtained from at least one of the directory servers DS; DS1-DSn and/or retrieving corresponding application data packets.”), 9:42-48 (“The obtained addresses IS; IS1-ISp and optionally other information can be returned to the access server AS; AS1-ASr, which...retrieves the specified application data packets from the identified application servers ...”), 14:7-11 (“[T]he directory server storing the object data determines from the object data application server addresses IS; IS1-ISp, storing application data packets associated with the object and returns retrieved address and application identifiers to the access server AS; AS1-ASr.”).) The second step involves using the object data to retrieve application data packets from the application servers, or alternatively, transmitting the object data directly to the user (e.g., a mobile telephone or data processing unit). (*Id.*, 8:32-36.)

B. Vingralek

41. 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.”).)

42. 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.*)

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

C. Krasner

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

45. In particular, Krasner discloses a “system for tracking the location of a mobile GPS receiver over time as it moves.” (*Id.*, 3:56-57.) Krasner describes its system as an improvement upon similar prior art systems which allow for tracking the position of a mobile GPS unit (e.g., the GPS receiver of a cellular telephone or a car’s mobile telephone) over time. (*Id.*, 1:48-51 (“Recently, GPS receivers have been used with radio transmitters, such as a cellular telephone or a mobile telephone in a car to transmit the position of the receiver as it moves.”), 2:39-41 (“[I]t is desirable to provide an improved method and system for providing multiple sets of position information over a period of time through a mobile GPS

unit.”).) Krasner explains that the mobile GPS unit receives GPS signals at various positions and determines pseudoranges from those signals, which it stores along with a corresponding time stamp in its own memory. (*Id.*, 5:49-53.) Krasner’s system also includes a location server (also called a basestation server) which is in wireless communication with the mobile GPS unit that is being tracked. (*Id.*, 4:10-13.) Krasner explains that when a predetermined type event occurs, the mobile GPS unit may transmit the pseudoranges and corresponding time stamps to the location server via the wireless communication system. (*Id.*, 4:66-5:24 (“When the predetermined type event does occur,...the stored pseudoranges and the corresponding time stamps are transmitted via a wireless communication system...to the location server.”).) Examples of such predetermined type events include the mobile GPS unit having collected a certain number of signals (e.g., transfer to the location server occurs after a mobile GPS unit has collected seven sets of signals); the mobile GPS unit reaches a memory limit; a predetermined period of time has lapsed since the last transmission; or an emergency event is detected, such as a car accident, inflated airbag, or other alarm condition. (*Id.*, 5:49-6:15.)

46. After the pseudoranges and time stamps are transmitted to a location server, the mobile GPS unit may clear that portion of its memory to allow for

further collection of pseudoranges and corresponding time stamps. (*Id.*, 5:20-24 (“[T]he memory which stored the pseudoranges and the time stamps is cleared for that portion of the memory. This will permit another set of pseudoranges to be collected along with their corresponding time stamps and stored and later transmitted.”).) Krasner explains that the location server may “determine[] a position of the transmitting mobile GPS unit may apply a differential GPS correction (e.g., based on data from a GPS reference receiver) to the pseudoranges, and then “determine[] a position of the transmitting mobile GPS unit from each set of corrected pseudoranges and the corresponding time stamp” so that “the location server may be able to track the movement of [the] mobile unit in space and in time.” (*Id.*, 6:54-7:23.) The location server may use this information to “provide concierge services or routing information” to a user through the wireless communication system. (*Id.*, 7:23-28.) Krasner discloses a wireless communication system in which a location server stores geographical location information of a mobile GPS unit that may be provided to a user.

47. Krasner discloses that its method provides the benefits of prior approaches, e.g., by “provid[ing] a way to track the route of a moving GPS receiver” (*id.*, 2:10-12), but offers a number of advantages over prior art systems. (*Id.*, 2:10-41, 5:25-48.) For example, Krasner’s method offers “a number of

advantages over the prior art technique of [the mobile GPS unit] determining a position at each point and then transmitting these positions” to the location server because unlike the prior art technique, Krasner’s method does not require an adequate view of the sky or an ability for the mobile unit to read satellite ephemeris data. (*Id.*, 5:25-48; *see also id.*, 2:10-30.) Krasner’s method is also preferable to prior art methods in which the GPS transmits pseudoranges to a location server one at a time because it would offer battery life benefits and reduce congestion in the wireless communication link. (*Id.*, 2:31-38.)

D. Sato

48. Sato relates to a method of partitioning servers in a distributed environment. For example, Sato describes a “client-server system built on a distributed system” in which a client “issue[s] a service request to a specific server and receive[s] a service.” (Ex. 1011, ¶[0003].) Sato explains that the system uses another set of servers, i.e., service location servers, which provide location information about the location of the specific server that provide services to the client. (*Id.*, ¶[0003] (“One method to efficiently investigate the existence of a server that provides the service requested by a client and to efficiently obtain information for establishing the communicating means with the server is to use a server that services location information of various servers (hereinafter referred to

as a service location server).”).) Sato further describes “the location information of each server is one of the global system resources,” and discloses a “server partitioning method that can partition the service range and manage location information in a distributed manner when the load on the service location server exceeds a predetermined threshold,” rather than having to pre-design how to manage the location information on each service location server. (*Id.*, ¶[0005].)

49. Sato explains that each “service location server” includes a “load measuring means” that measures the service load status of the server by measuring various “service load information.” (*Id.*, ¶[0009].) The measured “service load information...includes [1] the maximum number of service executions per unit time...after starting the service of the server..., [2] reference time...for determining whether or not the measurement time has passed, [3] a counter...for the number of services executed by the service location server since the reference time...was set, and [4] a measurement time interval.” (*Id.*, ¶[0012].) Sato discloses that “[t]he maximum number of service executions...is set by the load measuring means.” (*Id.*) Sato explains that the system implements “partition criteria information...[which] has a predetermined threshold value...for the maximum number of service executions per unit time.” (*Id.*, ¶[0013].) “Whether or not to partition the server...is determined by comparing the maximum number

of service executions...with the threshold value.” (*Id.*) For example, “[i]f the value of the maximum number of service executions...is greater than the threshold value..., it is determined that partitioning is necessary, and otherwise, it is determined that partitioning is not necessary.” (*Id.*, ¶[0029].) As Sato explains, if it is determined that partitioning is necessary, a server partition processing step is then executed in which information is migrated to another server. (*Id.*, ¶¶[0029]-[0030].)

VIII. CLAIM CONSTRUCTION

50. 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 August 13, 1999) having taken into consideration the language of the claims, the specification, and the prosecution history of the '978 patent. I have also been asked to consider and apply in my analysis the interpretations of certain terms that I understand are from a prior IPR proceeding and a district court proceeding, neither of which involves Petitioner. I

have applied these understandings in my analysis and in forming my opinions where appropriate in this Declaration.³

³ 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.)

A. “identifier”

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

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

B. “location information”

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

“one or more identifiers and their associated locations”

C. “location server”

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

“a network-attached component that maintains a set of identifier/location mappings that are modified or returned in response to location request messages from clients.”

D. “client”

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

“a network-attached component (which may be software or hardware) that initiates update or lookup of identifier/location mappings from a location server with location request messages.”

E. “plurality of location servers”

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

“location servers in non-hierarchical structures.”

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

'978 Patent Terms	Interpretation from Other Proceedings
“identifier”	“a unique encoding that identifies an individual entity, and with which zero or more data location specifiers are associated in a location server”
“location information”	“one or more identifiers and their associated locations”

'978 Patent Terms	Interpretation from Other Proceedings
“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”
“plurality of location servers” ⁴	“location servers in non-hierarchical structures”

⁴ I also apply this interpretation to claims reciting “plurality of data location servers.”

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

57. In my opinion, the prior art discussed below discloses and/or suggests the limitations of claims 10-17, 23-26, and 30.⁵

A. Skagerwall in View of Vingralek Discloses and/or Suggests Claims 10, 12-17, 23, and 30

1. Claim 10

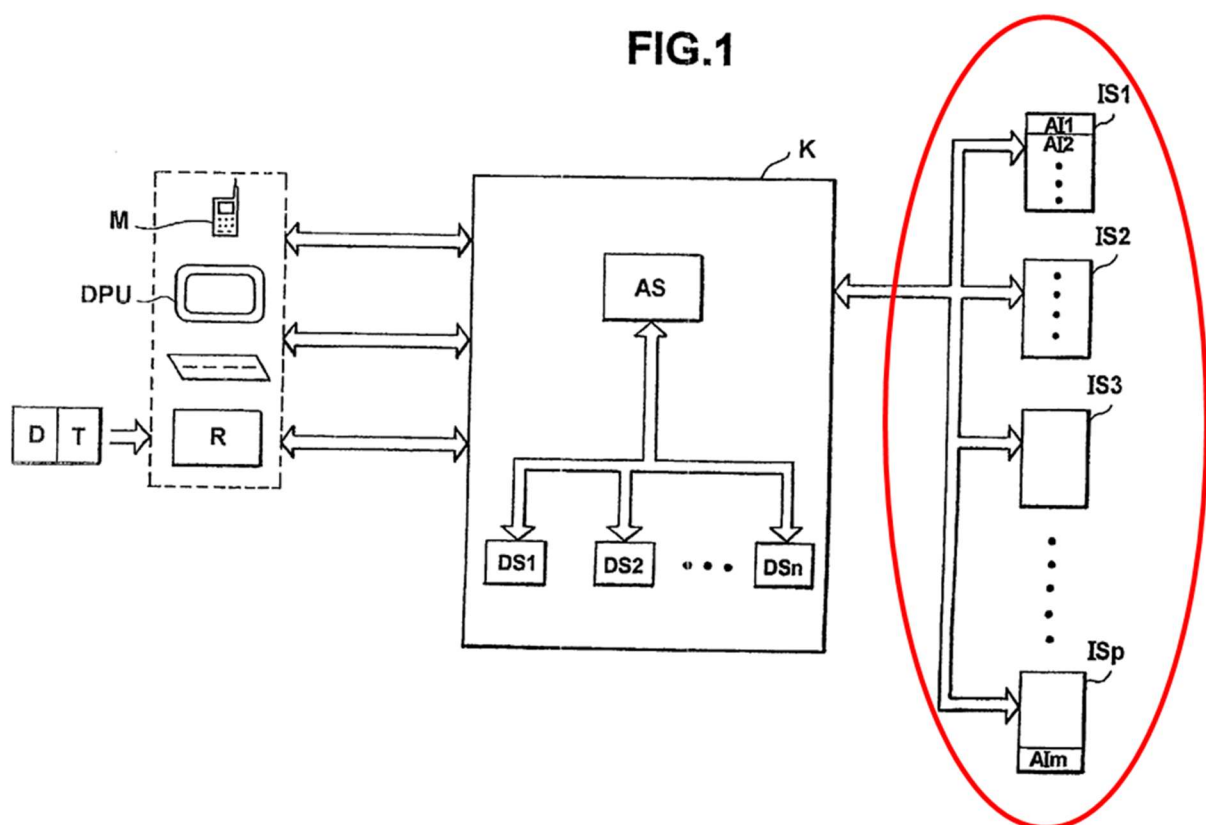
- i) [10.pre] “A system having a plurality of location servers for managing location information and providing location information to location queries, the system comprising:”**

58. 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, Skagerwall discloses the limitations therein. Skagerwall discloses a system that links objects, e.g., products in the physical world, to different types of information related to the objects. (Ex. 1006, Abstract (“[c]ommunication system and method for linking each of a plurality of tagged object[s] to information

⁵ 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.

services stored on a network”), 6:45-48 (“[t]he system is arranged to link the plurality of objects, which can, for example, be any products or items of the physical world to various kinds of information related to the objects”).) The system links objects to information or application data packets about those objects. (*Id.*, 4:1-5 (“The invention advantageously allows to link an arbitrary number of objects of the real world to information services stored on an arbitrary number of data processing units or application servers, which may be located at arbitrary places.”), 4:6-11 (“The data processing units or application servers preferably store a plurality of application data packets comprising data related to the services. Each of the plurality of application data packets may be associated with one of the plurality of objects of the real world, i.e., it contains information related to the object.”).) The application data packets may include text, video, or programs. (*Id.*, 4:5-9 (“The services may be text, images, application programs or any other kind of messages.”), 7:13-17 (“Data stored on the application servers can, for example, be text, such as a user manual of a product, links to other servers storing related data or other kind of written information. Further, the data can comprise images, video, audio signals and the like.”).) Skagerwall discloses with respect to Figure 1 that the application data packets, each identified by an application identifier (AI1-AIm), are distributed across application servers on a network, each identified by an

address (IS1-ISp). (*Id.*, 6:34-37 (“The system kernel K is arranged to communicate data between a plurality of application servers and a telephone M, a data processing unit DPU or the like. The application servers each have an address IS; IS1-ISp and store a plurality of application data packets, each having an application identifier AI1-Aim.”), 7:4-8 (“The plurality of application servers with the addresses IS; IS1-ISp and storing data related to the objects D may be placed at arbitrary locations in the world, similar to today’s computer networks, such as intranets, company networks, or the Internet.”).)

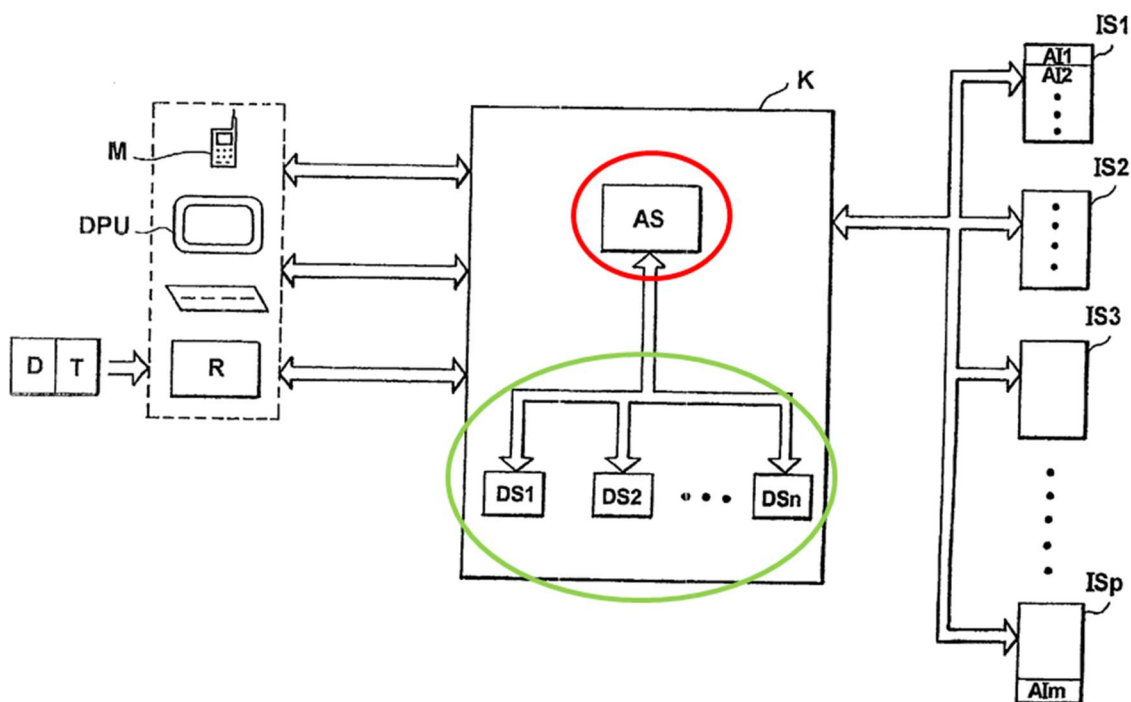


(*Id.*, FIG. 1.)

59. The system includes “a plurality of location servers for managing location information.” Skagerwall discloses with respect to Figure 1 that the system includes an access server (red) and a network of directory servers (green) (“a plurality of location servers”). (*Id.*, 6:32-34 (“The embodiment of FIG. 1 comprises a system kernel K with at least one [access] server AS; AS1-ASr and a plurality of directory servers DS; DS1-DSn.”).)⁶

⁶ A person of ordinary skill in the art would have understood that the reference to “server AS; AS1-ASr” as “application server[s]” is a typographical error, as these are referred to as “access server[s]” elsewhere in the specification, while the servers IS; IS1-ISp are referred to as the addresses of “application servers.” (*See, e.g.*, Ex. 1006, 9:1-9.)

FIG.1



(*Id.*, FIG. 1.) Each directory server (“location server”) stores object data, which associates objects with application data packets, and is involved in the retrieval of application data packets from application servers. (*Id.*, 4:13-16 (“[E]ach of a plurality of directory servers stores part of an arbitrarily scaleable data base containing object data associating each of the objects with at least one of the application data packets. The plurality of directory servers serve as [a] key resource in retrieving an application data packet related to an object.”), 8:19-22 (“Each of the directory servers DS; DS1-DSn preferably stores object data related to a small number of the range of available object tags T; T1-Tw.”), FIG. 1; *see*

also id., Abstract (“The system reads an object tag of one of the objects and retrieves information about the object stored on storage located at arbitrary places.”).) Skagerwall explains that each directory server (“location server”) stores object data, including object tags, associated application identifiers of application data packets, and addresses of the application servers storing those packets (collectively “location information”), and that the object data provide information about where an application data packet is stored. (*Id.*, 8:22-27 (“[O]bject data related to an object tag preferably includes application identifiers A11-A1m of application data packets related to the object as well as information about storage locations of the application data packets, e.g., the addresses of application servers IS; IS1-ISp storing the application data packets.”), 15:1-52 (discussing the storage of object tags, application identifiers, application server addresses on a directory server, and explaining how this information characterizes the location of a given application data packet on a network of application servers), 19:35-39 (“the directory servers store object data associated with the assigned object tags in a distributed data base, the object data associating each of said objects with at least one of said application data packets”), 19:51-53 (“object data corresponding to an object comprise an object tag, at least one application identifier and the address of the at least one application server storing the at least one application data packet

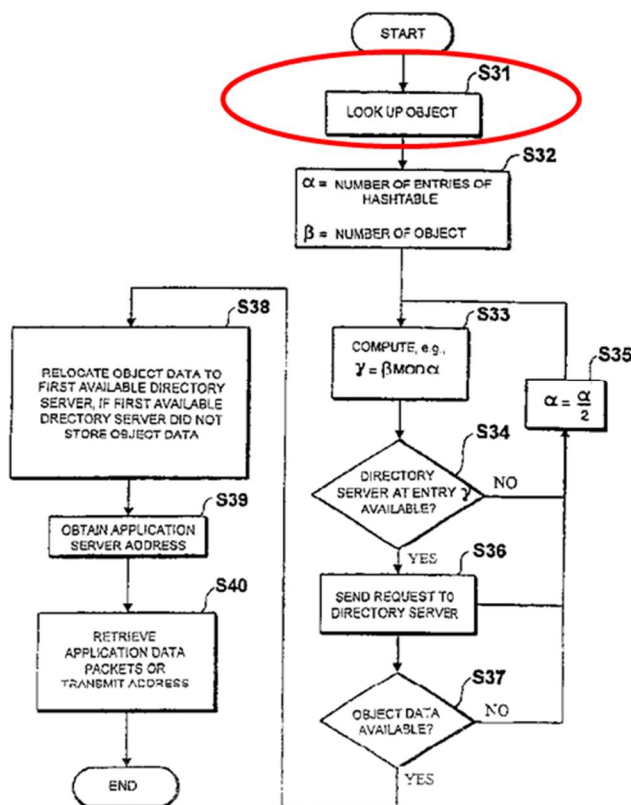
identified by the at least one application identifier”).) A person of ordinary skill in the art would have understood that the directory servers (“plurality of location servers”) manage the object tags, application identifiers, and application server addresses (“for managing location information”) since they are responsible for storing such information.

60. Skagerwall also discloses that the “plurality of location servers” are for “providing location information to location queries.” Skagerwall discloses a two-step process for retrieving application data packets, where the first step comprises identifying a directory server storing object data relating to an object and retrieving that object data, and the second step involves using the object data to retrieve application data packets from the application servers. (*Id.*, 8:22-36.) The retrieval process is further explained below.

61. Skagerwall discloses that the system kernel includes an access server that receives an information request about an object (step S31 of Figure 3 below), where the request at least includes an object tag. (*Id.*, 6:41-43 (“A plurality of objects D is marked with a plurality of tags T... A reader R is provided for reading data of an object tag and transmitting same [to] the system kernel K.”), 9:1-5 (“[A]t least one access server AS; AS1-ASr...is responsible for receiving a request for information about an object including an object tag T; T1-Tw read by the

reader R.”), 13:23-26 (“In a step S31 the system kernel K receives a request from a user, requesting information related to an object. A request at least includes an object tag T; T1-Tw of an object D...”), 13:29-30 (“Preferably, the request is received by one of the access servers AS; AS1-ASr.”), FIG. 3; *see also id.*, FIG. 2a (displaying in step S24 a lookup request for object data related to an object tag), 12:27-32 (describing the lookup request in step S24 of Figure 2a).)

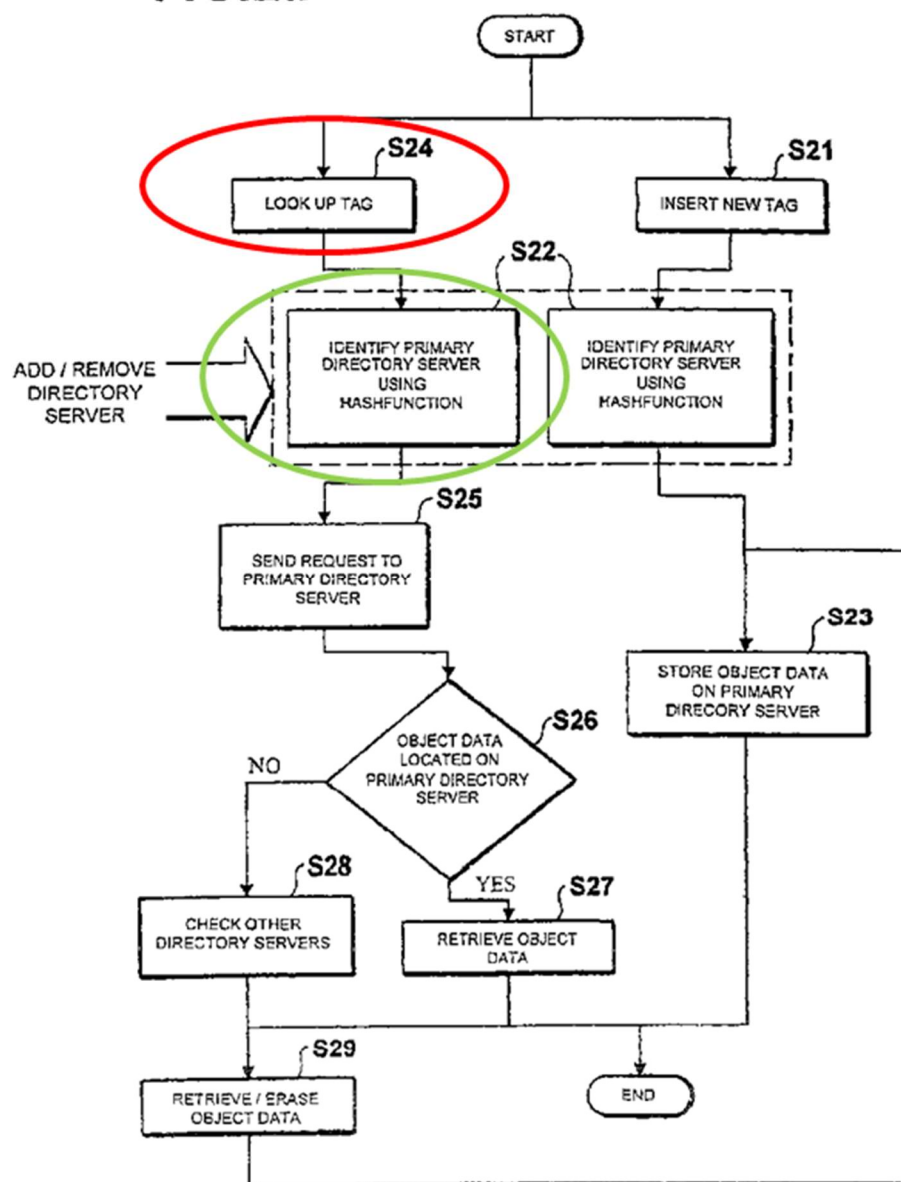
FIG.3



(*Id.*, FIG. 3 (annotated).)

62. Skagerwall discloses that upon receiving the information request, the access server performs operations to identify the directory server (“location server”) that stores object data concerning the location of application data packets. (*Id.*, 9:18-22 (“After receiving a request for information including the object tag T, the access server AS; AS1-Asr performs operations to identify the directory server DS; DS1-Dsn storing information concerning application data packets related to an object.”).) As shown in Figure 2a, after receiving an information request including an object tag in step S24 (red), a primary directory server (“location server”) is determined in step S22 (green) using a hash function. (*Id.*, 12:30-37.) In particular, the hash function is applied to the object tag from the information request, with the result serving as an index to search a table listing the available directory servers. (*Id.*, 12:8-12 (“In a step S22 the primary directory server...is computed based on, e.g., a number assigned to the new object tag as well as based on the number of available directory servers”); *see also id.*, 10:28-11:16, 13:15-57 (describing the application of a hash function to an object tag, with the result serving as an index to a table of directory servers).)

FIG.2a



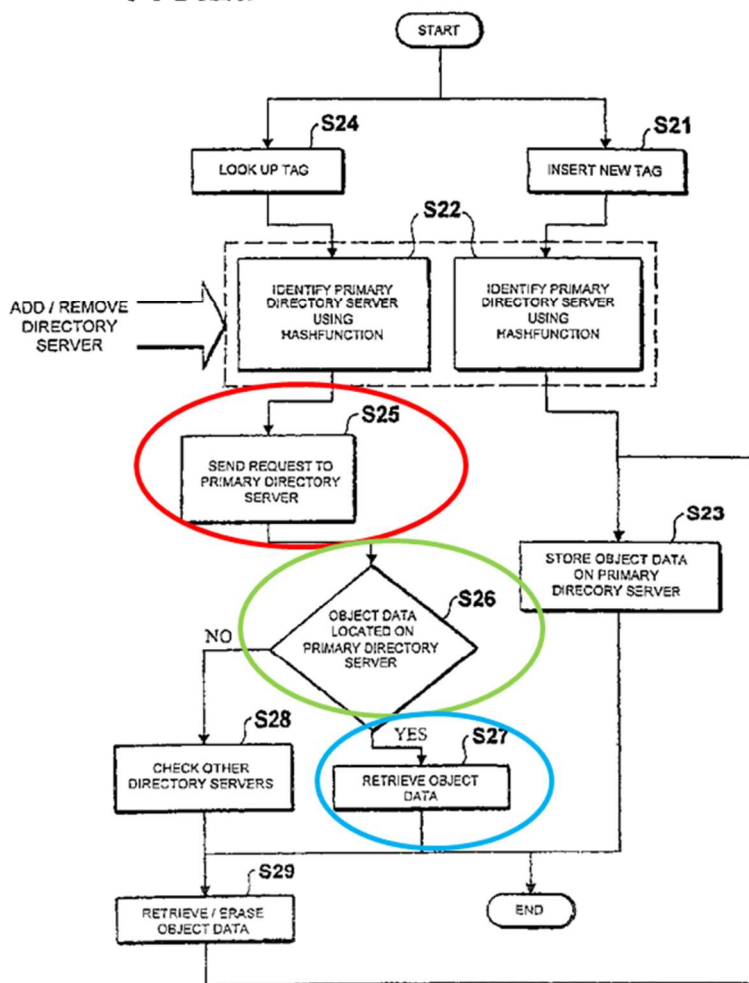
(*Id.*, FIG. 2a (annotated).)

63. In step S25 (red in Figure 2a below), the access server then sends a request (“location quer[y]”) that includes the object tag to the first directory server

(“location server”) to retrieve object tags, application server addresses, and application identifiers (“providing location information to location queries”). (*Id.*, 9:9-13 (“[T]he at least one access server AS; AS1-Asr executes the chain of requests connected to retrieving information from the system with respect to the object tag T; T1-Tw....”), 9:23-25 (“[The access server] send[s] the request including the object tag T; T1-Tw to one of the directory servers DS1; DS2; ...; DSn.”), 19:42-46 (“retrieving the object data corresponding to the object tag”), 19:51-55 (“object data corresponding to an object comprise an object tag, at least one application identifier and the address of the at least one application server storing the at least one application data packet identified by the at least one application identifier”); *see also id.*, 9:42-48 (“The obtained addresses IS; IS1-ISp and optionally other information can be returned to the access server AS; AS1-ASr, which...retrieves the specified application data packets from the identified application servers”), 14:7-11 (“[T]he directory server storing the object data determines from the object data application server addresses IS; IS1-ISp, storing application data packets associated with the object and returns retrieved address and application identifiers to the access server AS; AS1-ASr.”), 12:37-40 (access server transmitting a request for object data to the primary directory server in step S25), FIG. 2a (step S25).) In step S26 (green), the primary directory server

(“location server”) determines whether it contains the object data associated with the object tag. (*Id.*, 12:41-43; *see also id.*, 8:22-27.) If the directory server contains the object data, then in step S27 (blue), that object data (object tag, application server addresses, and application identifiers) is retrieved and subsequent steps for providing the user with the desired information are performed (e.g., retrieving application data packets from application servers based on the object data). (*Id.*, 12:54-62.)

FIG.2a



(*Id.*, FIG. 2a (annotated).)

64. Skagerwall also discloses this limitation as interpreted in the Other Proceedings. Skagerwall discloses the claimed “location server[]” under the Interpretation from Other Proceedings, i.e., “a network-attached component that maintains a set of identifier/location mappings that are modified or returned in response to location request messages from clients.” (*See* above Section VIII; *see*

also Ex. 1001, 4:34-38.) In particular, Skagerwall discloses that each directory server is a component attached to a distributed network (i.e., a network-attached component). (Ex. 1006, 8:15-19 (“the data base of object data is distributed over a plurality of directory servers”), 16:34-36 (all components in the system may be “connected via a network N for communication with each other”), FIG. 1 (depicting a network of directory servers DS; DS1-DSn).) As discussed for limitation [10.b], each object tag is a claimed “identifier” that identifies an object, under the Interpretation from Other Proceedings of “identifier.” (Section IX.A.1.iii.) Skagerwall discloses application identifiers of application data packets and addresses of the application servers storing those packets are both locations which together identify the location an application data packet is stored. (Ex. 1006, 8:22-27, 15:1-52.) Indeed, the application server addresses and application identifiers both disclose “location[s]” (or similar terms) as described in the ’978 patent. (Ex. 1001, 4:28-30.) A person of ordinary skill in the art would have also understood that each directory server maintains mappings from object tags (i.e., identifiers) to application server addresses and application identifiers (i.e., locations of application servers storing application data packets) because Skagerwall discloses that object tags and associated object data are distributed across directory servers based on a hash function, and that each directory server

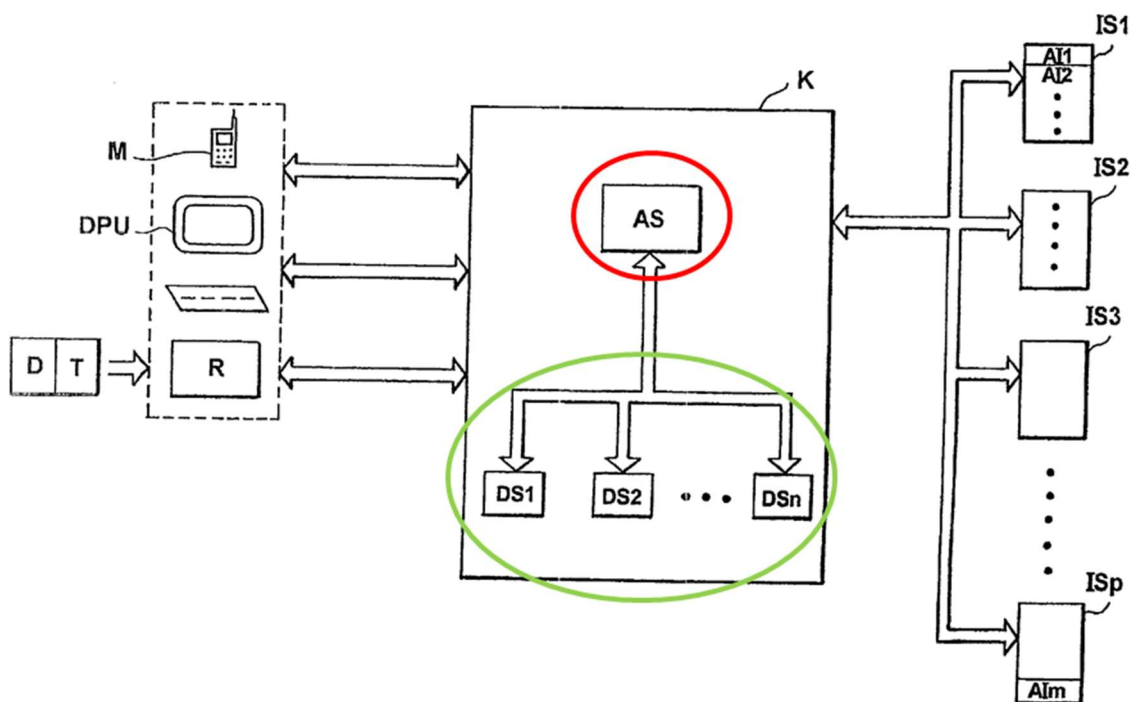
has a memory means associating object tags with corresponding object data (including application server addresses and application identifiers). (Ex. 1006, 8:43-63 (“[T]he hashfunction is used to assign object data related to a group of objects to each of the directory servers. Thus, each of the object tags is assigned to a target directory server using the hashfunction....”), 15:15-20 (“[D]irectory server DS1 further comprises address memory means AM for storing a plurality of address data blocks AB1-ABk, each comprising one of the application identifiers AI1-Aim, the corresponding application server address IS; IS1-ISp and, optionally, corresponding object tags T; T1-Tw.”), 15:49-52 (“[T]he directory server may comprise single memory means, in which case all object data related to an object are stored in the provided memory means.”).)

65. Further, Skagerwall discloses that the directory server returns identifier/location mappings in response to a request from the access server for location information associated with an object tag (i.e., in response to location request messages from a client). For example, as discussed above and for limitation [10.b], the object tag, application server addresses, and application identifiers (i.e., identifier/location mappings) are returned to the access server if the directory server determines it contains the requested location information. (Section IX.A.1.iii; Ex. 1006, 19:42-46, 19:51-55; Ex. 1001, 2:32-37, 4:61-64.) Skagerwall

further discloses that identifier/location mappings may be updated/modified in response to a request from the access server to add new location information (i.e., another type of location request message from the client). (Ex. 1006, 12:3-26, 14:29-56; Ex. 1001, 10:1-4.) Thus, Skagerwall discloses the claimed “location server[]” under the Interpretation from Other Proceedings.

66. Further, Skagerwall discloses the claimed “plurality of location servers” under the Interpretation from Other Proceedings, 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 directory servers (each a “location server”) are arranged in a flat, cluster topology, rather than a hierarchical or tree-like structure where servers can only communicate with nodes immediately above or below it in the tree, because—as shown below in Figure 1—there is no hierarchy of intermediary servers between the access server (in red) and the directory servers (in green), and the directory servers themselves are not arranged in a hierarchy.

FIG.1



(Ex. 1006, FIG. 1 (annotated).) Further, Skagerwall discloses that the hash function is used to distribute object tags and associated object data (e.g., application server addresses) among the directory servers. (*Id.*, 8:15-27, 8:38-54, 8:55-59.) A person of ordinary skill in the art would have thus understood that because a hash function is used for distribution across directory servers, Skagerwall's directory servers are arranged in a non-hierarchical configuration.

67. Skagerwall also discloses the claimed "location information" under the Interpretation from Other Proceedings, i.e., "one or more identifiers and their associated locations." (*See above Section VIII.*) As discussed above, the object

tags stored on a directory server are identifiers of objects, and the application server addresses and application identifiers are both locations associated with an object tag.

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

68. Skagerwall discloses and/or suggests this limitation. **First**, Skagerwall discloses “a plurality of location servers containing location information corresponding to a plurality of entities.” As discussed for limitation [10.pre], a network of directory servers (“a plurality of location servers”) that contain object tags, associated application identifiers, and associated application server addresses (collectively “location information”), which correspond to multiple objects (“corresponding to a plurality of entities”). (See Section IX.A.1.i; Ex. 1006, 8:15-27 (discussing the storage of object tags and associated object data (e.g., application identifiers and application server addresses) for multiple objects on the plurality of directory servers).)

69. **Second**, Skagerwall discloses and/or suggests “the location information formatted according to a transfer protocol configured for manipulating location information.” A person of ordinary skill in the art would have found it beneficial to format communications sent between an access server and directory server according to a transfer protocol, e.g., hypertext transfer protocol (HTTP). Skagerwall discloses that its system may “use standard and/or amended protocols of a known datacom- or telecom network for retrieving information associated with an object.” (Ex. 1006, 9:59-62.) Skagerwall also discloses that the system may be “used in combination with a known datacom- or telecom network, such as the Internet,” and that at least lookup procedures may be performed “via a HTTP request.” (*Id.*, 18:2-7, 18:24-26.) A person of ordinary skill in the art would have understood HTTP is a transfer protocol broadly governing communications between web clients and web servers, and its application is not limited to communications occurring during lookup procedures. Moreover, a person of ordinary skill in the art would have found it beneficial to use a transfer protocol compatible with Skagerwall’s system, e.g., HTTP, for not only the lookup processes, but also other processes involving the manipulation (e.g., storage, retrieval, etc.) of object data to enable the system to operate using a widely-used, Internet-compatible, and well-known protocol to improve reliability. Thus, a

person of ordinary skill in the art would have been motivated to format communications (e.g., requests) between an access server and directory server, and the storage/retrieval of the location information on a directory server (“manipulating location information”), in accordance with HTTP (“the location information formatted according to a transfer protocol configured for manipulating location information”).

70. A person of ordinary skill in the art would have had the skill and reasonable expectation of success in implementing this feature because it would have involved modifying the system in a manner that Skagerwall suggests is feasible—implementing the system in accordance with a standard protocol, e.g., HTTP. Moreover, the modification would have involved applications of known technologies (e.g., known communication systems of Skagerwall) according to known methods (e.g., known transfer protocol used in Skagerwall or HTTP) to yield the predictable result of a system having communications and processes for the manipulation of the disclosed “location information” formatted according to HTTP, as discussed above.

71. **Third**, Skagerwall discloses “the location information...comprising at least one application server address.” As discussed for limitation [10.pre], the

“location information” includes application server addresses (“at least one application server address”). (*See* Section IX.A.1.i.)

72. **Fourth**, Skagerwall discloses “the plurality of location servers are arranged in a cluster topology such that each location server contains a unique set of location information of an aggregate set of the location information.” As discussed for limitation [10.pre], a person of ordinary skill in the art would have understood that the directory servers (“the plurality of location servers”) are arranged in a cluster structure (“arranged in a cluster topology”), rather than a hierarchical or tree-like structure. (*See* Section IX.A.1.i.) Skagerwall further discloses that the database of object tags and associated object data (including application server addresses and application identifiers) is distributed evenly across multiple directory servers, such that each directory server stores object data related to certain object tags. (Ex. 1006, 8:15-27, 8:48-51 (“[E]ach of the object tags is assigned to a target directory server using the hashfunction. Preferably, the hashfunc[tion] distributes the object tags evenly over the available directory servers.”).) A person of ordinary skill in the art would have thus understood that each directory server (“location server”) contains a unique set of object tags, application server addresses, and application identifiers (collectively “location

information”) out of the full set of all the object tags, application server addresses, and application identifiers (“an aggregate set of the location information”).

- iii) **[10.b] “programming logic stored on each of the plurality of location servers responsive to a location query for a desired identifier to return one of a location message, wherein a queried location server returns a location message if the queried location server contains location information for the desired identifier, and a redirect message if the queried location server does not contain location information relevant to the desired identifier, wherein the redirect message comprises information for finding a location server having location information related to the desired identifier.”**

73. Skagerwall in view of Vingralek discloses and/or suggests this limitation. **First**, Skagerwall discloses “programming logic stored on each of the plurality of location servers” that is configured to perform operations performed by a “location server” discussed herein. Skagerwall discloses that each directory server (“each of the plurality of location servers”) has a central processing unit that controls the operations of the server. (Ex. 1006, 15:1-7 (“The directory server DS1 comprises a central processing unit CPU for controlling operations of the directory server.”).) Thus, each directory server would store programming logic (“programming logic stored on each of the plurality of location servers”) to perform operations thereof, including those recited in limitation [10.b] and discussed below.

74. **Second**, Skagerwall discloses “programming logic...responsive to a location query for a desired identifier to return one of a location message, wherein a queried location server returns a location message if the queried location server contains location information for the desired identifier.” As discussed for limitation [10.pre], the primary directory server (“queried location server”) receives a request (“location query”) from the access server for information concerning the location of application data packets related to an object tag (“for a desired identifier”), where that request includes the corresponding object tag. (*See* Section IX.A.1.i.) The primary directory server then determines whether it stores the mapping of an object tag to addresses of application servers and application identifiers (the three types of information collectively “location information”) for the object tag included in the request (“for a desired identifier”). (*See id.*) If the directory server (“queried location server”) contains the disclosed “location information” associated with the object tag (“for a desired identifier”), the disclosed “location information” is retrieved and subsequent steps for providing the user with the desired information are performed. (*See id.*) For example, as discussed for limitation [10.pre], Skagerwall discloses that the disclosed “location information” may be sent to the access server (“returns a location message”), which the access server may use to determine where to find the application data

packets associated with an object tag. (*See id.*) A person of ordinary skill in the art would have understood that the “location information” is returned in a message because Skagerwall discloses that other communications from directory server to access server are “message[s].” (Ex. 1006, 13:5-6, 13:64-65.)

75. The object tag is an “identifier” under the Interpretation from Other Proceedings, i.e., “a unique encoding that identifies an individual entity, and with which zero or more data location specifiers are associated in a location server.” (*See* above Section VIII; *see also* Ex. 1001, 4:25-28.) A person of ordinary skill in the art would have understood that the object tag is an encoding that uniquely identifies an object (i.e., an individual entity) because, as discussed for limitation [10.pre], the tag is included in a query to identify the specific object for which information is sought. (*See* Section IX.A.1.i; *see also* Ex. 1006, 3:4-5 (“a plurality of objects (D) each identified by an object tag (T)”), 7:21-25 (object tag may be read from a barcode), 16:5-9 (“The bits [of an object tag] may code digits or signs....”).) As further discussed, zero or more application identifiers and application server addresses (both data location specifiers), which specify the location of application data packets (i.e., data) are associated with the object tag in a directory server (i.e., a location server, as interpreted in the Other Proceedings). (*See* Section IX.A.1.i; Ex. 1006, 8:15-27, 15:4-52.)

76. **Third**, Skagerwall discloses that “if the queried location server does not contain location information relevant to the desired identifier,” the directory server (“queried location server”) is configured to generate “information for finding a location server having location information related to the desired identifier.” As discussed for limitation [10.pre] with respect to Figure 2a, a primary directory server is determined in step S22 based on the application of a hash function to an object tag, with the result used as an index for searching the hash table. (Section IX.A.1.i.) As further discussed therein, in step S26 the primary directory server (“queried location server”) determines whether it stores the mapping of an object tag to application server addresses and application identifiers (the three types of information collectively “location information”) for the object tag included in the request (“relevant to the desired identifier”). (*See id.*) If the queried server includes the requested information, then the disclosed “location information” is retrieved by the access server in step S27. (Section IX.A.1.i.) If the directory server determines in step S26 that it does not contain the requested information (“if the queried location server does not contain location information relevant to the desired identifier”), then the request is forwarded to another directory server in step S28, which is determined based on the application of the hash function. (Ex. 1006, 12:63-13:5.)

77. Skagerwall discloses that the primary directory server (“queried location server”) may be the component that determines the location of a “correct” directory server (i.e., the directory server that has the requested application server addresses and application identifiers associated with the object tag) using the hash function. (*Id.*, 9:18-28 (disclosing that the access server “send[s] the request including the object tag T; T1-Tw to one of the directory servers DS1; DS2;...; DS_n, which in turn, using the hashfunction and hashtable H, determines the directory server for storing the object tag T; T1-Tw and forwards the request to that particular directory server”), 10:22-25 (“[I]t is preferred, that one of the directory servers DS1; DS2;...; DS_n identifies the directory servers storing information related to a requested object tag.”).)

78. Thus, Skagerwall discloses that “if the queried location server does not contain location information relevant to the desired identifier,” the queried directory server (“queried location server”) is configured to generate “information for finding a location server having location information related to the desired identifier.”

79. I have been asked to consider a scenario where Skagerwall does not explicitly disclose “**programming logic...responsive to a location query for a desired identifier to return...a redirect message** if the queried location server

does not contain location information relevant to the desired identifier, **wherein the redirect message comprises information for finding a location server having location information related to the desired identifier.**” In that scenario, a person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek.

80. Like Skagerwall, 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 that are 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 Skagerwall, which discloses that the database of object data is distributed based on a hash function over a plurality of directory servers, so that the number of objects may be scaled by adding new directory servers. (Ex. 1006, 8:15-19, 8:43-54; 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 Skagerwall's system/method.

81. Vingralek additionally discloses propagating updates to clients' and other servers' address tables in a deferred manner. (Ex. 1007, 6.) To limit communication overhead, clients learn about changes to the address table only when a client request reaches a wrong server, in which case the server forwards the client's request to the correct server and provides updates to the client's address table. (*Id.*, 5 ("To limit the communication overhead, clients learn about bucket splits and changes to the server-locating hash function only when a client request is directed to the 'wrong' server," in which case, "the server forwards the client's request to the proper server, and notifies the client in order to update the client's stale information on the hash function."), 6 ("Similarly to [the] LH* [approach], updates to the address table are propagated to clients and other servers in a deferred manner; forwarding is used to locate buckets when a client has a stale address table.")) Thus, Vingralek discloses updates to a client's stale address table are deferred until a wrong first server is reached and the location of a second server needs to be calculated.

82. Vingralek's method of deferred address table updates operates as follows. A hash function is applied to a key, which is used as an index to an

address table listing available servers. (*Id.*, 9.) Each client and server has its own copy of the address table, but, for example, the client's 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).”); *see also id.*, 11-13.) When a client seeks to retrieve records associated with a given key, it applies the hash function to the key, identifies the server where that key is located by using the hashed key as an index into the current copy of the address table, and then sends the key in a request to that first server. (*Id.*, 9-10.) The first server that receives the key also applies the hash function to the key and identifies in its copy of the address table the server where the key resides (at least according to that first server's address table). (*Id.*, 10-11.) If the key was indeed sent to the correct server (i.e., if the identified server is the same as that first server), then that first 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 (i.e., the one identified by the above-described hash function application), and sends a message to the client to update the client's address table. (*Id.*; *see also id.*, 25 (referring to “client requests and server responses” as “messages”).)

83. The key is eventually received by the correct server and the request from the client is executed. (*Id.*, 11.) Via the address table update, “clients gradually bring their perception of the file up-to-date, so that the probability of wrongly addressed messages is significantly reduced.” (*Id.*) Vingralek explains that if a server undergoes changes (e.g., if information is moved to another server), the server receives a new address table. (*Id.*, 12-13.) Consistent with the above, a person of ordinary skill in the art would have understood that a server would have up-to-date information at least with respect to its own keys/information, such that the process of updating the client’s address table each time a request is forwarded to the next server (until the request reaches a “correct” server containing the requested information, i.e., “a location server known to have location information relevant to the location query”) would have gradually brought the client’s perception of the address table up-to-date. Thus, Vingralek discloses a first server calculating the location of a “correct” server to send a request to, and further discloses that if a first server does not have the requested information, a message (“redirect message”) is sent to the client with information to update the client’s address table.

84. Additionally, Vingralek discloses that “[t]he most salient feature of [the LH*] approach,” where a client’s hash table is updated in a deferred manner,

“is that its communication overhead is largely independent of the number of servers and clients in the system; thus, it is considered a scalable approach.” (Ex. 1007, 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.)

85. A person of ordinary skill in the art would have been motivated to include features similar to those disclosed by Vingralek when implementing Skagerwall by providing a deferred address/hash table update to the client in order to limit communication overhead associated with updating hash tables in a scalable system. Indeed, Skagerwall itself explains that, while an access server’s hash table is not always up to date, it is preferably not updated every time there is a change to the hash table, so as to conserve resources. (Ex. 1006, 10:12-20.) As discussed above, Vingralek’s method of updating the client’s address table only when a request reaches the wrong server (i.e., in a deferred manner) is beneficial because it limits communication overhead, and improves the scalability of the network because communication overhead is independent of number of servers and clients in the system. (Ex. 1007, 5.) Thus, a person of ordinary skill in the art would have

found it beneficial to include Vingralek's method of deferred hash table updates, via a redirect message, when implementing Skagerwall's system/process.

86. A person of ordinary skill in the art would have further understood that it would also be beneficial for Vingralek's method of updating the client's address table via redirect message would also be beneficial in order to provide the addresses of "correct" location servers to an access server. Skagerwall discloses that "[i]ntermediate steps are possible, where the look up request is relayed to further devices before it reaches the correct directory server." (Ex. 1006, 9:39-41.) Indeed, Skagerwall discloses an arrangement in which the first directory server ("queried location server") calculates the location of a correct directory server and the access server forwards the request to the correct directory server, but may involve an extra set of servers (referred to as broker servers), and does not explicitly disclose a specific process for transmitting the address of the correct directory server to the access server. (*Id.*, 16:47-51, 18:33-43.) Consistent with Vingralek's disclosure of updating the client's address table, a person of ordinary skill in the art would have been motivated to implement a system similar to that of Skagerwall where it includes in the redirect message only the portion of the address table corresponding to the location server(s) that contain the requested location information ("redirect message comprises information for finding a

location server having location information related to the desired identifier”). This would allow the access server to update the relevant portions of its own hash/address table, consistent with Vingralek’s teachings regarding the benefits of a deferred address table update, as well as provide the access server with the addresses of the location server(s) that should receive the forwarded request.

87. A person of ordinary skill in the art would have also understood that implementing Vingralek’s redirect message feature would have reduced communication bandwidth in several ways, consistent with Vingralek’s teachings of limiting communication overhead: by directly conveying the address of a correct directory server to the access server (without involving of an extra set of broker servers), and by including in the redirect message only the portion of the address table corresponding to the location server(s) that contain the requested location information, instead of the entire mapping/table. Such a modification would have been beneficial as it would have reduced the frequency of communications to a client and/or the use of the client’s computational resources, particularly in situations where the client has limited communication bandwidth and/or computational resources (e.g., resources for calculating the correct directory server). (Ex. 1007, 5, 7, 8; Ex. 1006, 5:28-37, 10:15-20, 17:1-5, 18:54-58.)

88. A person of ordinary skill in the art would have understood that this implementation, where a first directory server calculates a correct directory server and the access server forwards the request to that server (e.g. as disclosed by Skagerwall), would have been one of only several options, i.e., calculating and forwarding at either the access server or the directory server that processes the information request. Thus, a person of ordinary skill in the art would have been motivated to implement the process of calculating the addresses of correct directory server(s) at a directory server and transmitting a redirect message, including those addresses, to the access server for the access server to forward the request to that correct directory server. As such, a person of ordinary skill in the art would have been motivated to include features similar to Vingralek's "redirect message" features when implementing Skagerwall.

89. A person of ordinary skill in the art would have had a reasonable expectation of success combining the teachings of Skagerwall and Vingralek. As discussed above, both references are in the same field of art. (*See above.*) Furthermore, Skagerwall discloses an "error message" that is provided if none of the directory servers stores the requested object data (Ex. 1006, 13:5-6, 13:62-65), akin to the claimed "redirect message" that is provided if the location server does not contain the requested information. In fact, a person of ordinary skill in the art

would have appreciated that the concept of providing a message when requested information is not found was well known. (*Id.*, 2:40-43 (“[A] called address or link for retrieving a certain service from the network may no longer be valid and an error message is returned to the user.”).) Moreover, Skagerwall itself suggests that its system/process is flexible and additional processes may be included, particularly with respect to the process of relaying a lookup request to a correct directory server. (*Id.*, 9:39-41 (explaining that “[i]ntermediate steps” may be implemented when relaying a lookup request to a correct directory server).)

90. Additionally, a person of ordinary skill in the art would have understood that implementing the combination would have involved modifying the Skagerwall system in a manner that Skagerwall suggests is feasible—implementing a means of occasionally updating a hash table to reduce communication overhead, and implementing a means of providing addresses of correct directory servers to the access server. (*Id.*, 10:12-20.) A person of ordinary skill in the art would have also had the skill to implement and reasonable expectation of success in achieving such a modification, because it would have involved applications of known technologies (e.g., known systems of managing data and associated location information (Skagerwall)) according to known methods (e.g., known methods of occasionally updating hash tables using a

redirect message (Vingralek)) to yield the predictable result of a system that updates hash tables in a manner that limits communication overhead and allows for scalability, as discussed above. Accordingly, Skagerwall in view of Vingralek discloses and/or suggests this limitation.

91. Consistent with the above and in such a combined system, for example, the access server would apply the hash function to the object tag and its own copy of the hash table, and send a request to the primary directory server (“queried location server”), which would apply the hash function to the object tag and its own copy of the hash table in order to determine whether the server has the requested location information. If the primary directory server does not have the requested location information, e.g., application server addresses and application identifiers, associated with the object tag (“if the queried location server does not contain location information relevant to the desired identifier”), the directory server would calculate the location/address of another directory server that has the requested location information (“information for finding a location server having location information related to the desired identifier”). The directory server would then send a portion of the hash table corresponding to that directory server in a redirect message (“programming logic...responsive to a location query for a desired identifier to return...a redirect message...wherein the redirect message

comprises information for finding a location server having location information related to the desired identifier”). Such a redirect message (similar to Vingralek’s disclosed message to update a client’s hash table) would allow the access server to update its stale hash table, and additionally provide the access server with the address of a correct directory server, which the access server uses to find the correct directory server and forward the request to that server. (*Id.*)

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

92. Skagerwall discloses this limitation. As discussed for limitation [10.pre], Skagerwall discloses that the directory servers store object tags, application identifiers, and application server addresses (collectively “location information”), and each application server address indicates the location/address of an application server in a network (“location information comprises network address information”). (*See* Section IX.A.1.i.)

3. Claim 13: “The system of claim 12, wherein the network address information comprises a network address of a data repository containing data pertaining to the desired identifier.”

93. Skagerwall discloses this limitation. As discussed for limitation [10.pre] and claim 12, Skagerwall discloses that an application server address (“network address information”) indicates the network address of an application

server (“data repository”), which contains application data packets (“data”) pertaining to a specific object tag (“desired identifier”). (*See* Sections IX.A.1.i, IX.A.2.) A person of ordinary skill in the art would have understood that each application server is a “data repository” because it stores the application data packets (“data”) associated with tagged objects.

4. Claim 14

- i) **[14.pre] “A method of handling location queries in a network, the network comprising a plurality of location servers, each location server containing a unique set of location information of an aggregate set of location information correlating each of a plurality of identifiers with at least one location, the method comprising:”**

94. I understand that this limitation is a preamble of the claim and I have been asked to assume that the preamble of claim 14 is limiting. Under such an assumption, Skagerwall discloses this limitation. **First**, Skagerwall discloses “a method of handling location queries in a network, the network comprising a plurality of location servers.” As discussed for limitation [10.pre], Skagerwall discloses a system including a network of directory servers (“network comprising a plurality of location servers”), which store mappings of object tags, application server addresses, and application identifiers (collectively “location information”), which identify the location of application data packets. (*See* Section IX.A.1.i.) As

further discussed, Skagerwall discloses that the system performs a method of handling requests for application server addresses and application identifiers (“location queries”) pertaining to a specific object tag, where the request for that location information is sent to a directory server (“location queries in a network”). (See Section IX.A.1.i.)

95. **Second**, Skagerwall discloses “each location server containing a unique set of location information of an aggregate set of location information correlating each of a plurality of identifiers with at least one location.” Skagerwall discloses that each directory server (“location server”) stores “a unique set of location information of an aggregate set of location information” for the reasons discussed for limitation [10.a]. (See Section IX.A.1.ii.) As discussed for limitation [10.pre], the location information stored on each directory server is stored in identifier/location mappings, such that each of the object tags stored on a directory server (“each of a plurality of identifiers”) is associated with the corresponding application server addresses and application identifiers (together “at least one location”) for that object tag (“correlating each of a plurality of identifiers with at least one location”). (See Section IX.A.1.i.)

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

96. Skagerwall discloses this limitation. As discussed for limitation [10.pre], the access server (“client”) transmits the information request (“location query”) to a directory server (“receiving a location query from a client at one of the plurality of location servers”). (*See* Section IX.A.1.i.) As further discussed, the request (“location query”) is for the application server addresses and application identifiers associated with the object tag that identifies an object (“entity”) in the request, where those application server addresses and application identifiers together specify the location of the application data packets (i.e., data) of an object (“requesting an entity’s location”).

97. Skagerwall discloses the claimed “client” under the Interpretation from Other Proceedings, i.e., “a network-attached component (which may be software or hardware) that initiates update or lookup of identifier/location mappings from a location server with location request messages.” (*See* above Section VIII; *see also* Ex. 1001, 4:30-34.) A person of ordinary skill in the art would have recognized that the access server (“client”) is a hardware and/or software component attached to a network given that Skagerwall explains that all components in the system may be “connected via a network N for communication

with each other.” (Ex. 1006, 16:34-36.) As further discussed for limitation [10.pre], the access server sends a request (i.e., location request message) to a directory server (i.e., location server) to initiate lookup of mappings of object tags (i.e., identifiers) to application identifiers and application server addresses (which together provide location information about application data packets) from the directory server. (See Section IX.A.1.i.)

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

98. Skagerwall discloses this limitation. As discussed for limitation [10.pre], Skagerwall discloses a request (“query”) for the return of a mapping of an object tag to application server addresses and application identifiers (collectively “location information”) associated with an object (“entity”), where the request is sent to a primary directory server (“queried location server”) and includes the object tag that identifies an object (“entity identified in the query”). (See Section IX.A.1.i.) As discussed for limitation [10.b], if the directory server has the requested object tag, application server addresses, and application identifiers (“if the queried location server contains location information relevant to an entity

identified in the query”), the directory server returns a message to the access server (“sending a location response message to the client”) with the object tag, associated application server addresses, and associated application identifiers (“the location response message comprising location information”). (See Section IX.A.1.iii.) As further discussed for limitations [10.pre] and [10.b], this sent location information identifies the location of application data packets of an object, with the application server address identifying an application server that contains application data packets (i.e., data/information) associated with the object (“location information identifying at least one application server containing information relevant to the entity identified in the query”). (See Sections IX.A.1.i, IX.A.1.iii.)

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

99. The Skagerwall-Vingralek combination discloses and/or suggests this limitation. As discussed for limitation [10.pre], Skagerwall discloses a request (“query”) for object tags, application server addresses, and application identifiers

(collectively “data location information”)⁷ associated with an object (“entity”), where the request is sent from the access server to a primary directory server (“queried location server”) and includes the object tag that identifies an object (“entity identified in the query”). (See Section IX.A.1.i.) As discussed for limitation [10.b], the Skagerwall-Vingralek combination discloses and/or suggests that if the queried directory server (“queried location server”) does not have the object tags, application server addresses, and application identifiers (collectively “data location information”) that were requested (“if the queried location server does not contain data location information relevant to the entity identified in the query”), the directory server returns a message to the access server (“sending a redirect message to the client”) that identifies which of the other directory servers stores the requested application server addresses and application identifiers associated with the object tag (“the redirect message comprising information for finding a location server storing the entity identified in the query”). (See Section IX.A.1.iii.)

⁷ Skagerwall discloses “data location information” under the Interpretation from Other Proceedings for “location information” for the same reasons as discussed for limitation [10.pre]. (Sections VIII, IX.A.1.ii.)

5. **Claim 15: “The method of claim 14, further comprising applying information in the location query from the client to an indexing function to determine if the queried location server contains location information relevant to the entity identified in the query.”**

100. The Skagerwall-Vingralek combination discloses and/or suggests this limitation. As discussed for limitation [10.b], in the Skagerwall-Vingralek combination, a queried directory server (“queried location server”) applies the object tag identified in the request (“location query”) from the access server (“applying information in the location query from the client”) to a hash function (“indexing function”) to determine whether the directory server contains the object tag, application server addresses, and application identifiers mapping (collectively “location information”) associated with the object tag (identifying an object (“entity”)) included in the request (“to determine if the queried location server contains location information relevant to the entity identified in the query”). (*See* Section IX.A.1.iii.) A person of ordinary skill in the art would have understood that an indexing function is a function that returns the index to a table, and thus the hash function of the Skagerwall-Vingralek combination, which returns a value used as an index to the hash table, is an “indexing function.”

6. **Claim 16: “The method of claim 15, wherein each of the plurality of location servers comprises an indexed table of identifiers and associated locations, and wherein applying**

information in the location query comprises applying an identifier in the location query to a hash function.”

101. The Skagerwall-Vingralek combination discloses and/or suggests this limitation. Skagerwall discloses that each directory server (“each of the plurality of location servers”) has a memory means that stores object tags (“identifiers”) and the associated application server addresses and application identifiers (both “associated locations”). (Ex. 1006, 15:15-20 (“[D]irectory server DS1...further comprises address memory means AM for storing a plurality of address data blocks AB1-ABk, each comprising one of the application identifiers AI1-Aim, the corresponding application server address IS; IS1-ISp and, optionally, corresponding object tags T; T1-Tw.”), 15:49-51 (“[T]he directory server may comprise single memory means, in which case all object data related to an object are stored in the provided memory means.”).) A person of ordinary skill in the art would have understood that each directory server’s memory means comprises an “indexed table of identifiers and associated locations” because each object tag (“identifier”) is used as an index to retrieve all associated application server addresses and application identifiers (both “associated locations”). As discussed for claim 15, Skagerwall also discloses that “applying information in the location query” comprises applying an object tag (“identifier”) in the request (“location query”) to a hash function. (See Section IX.A.5.)

7. Claim 17

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

102. 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, Skagerwall in view of Vingralek discloses and/or suggests the limitations therein. Skagerwall discloses “a system having a plurality of location servers for storing and retrieving location information, wherein each of the plurality of location servers stores [a] unique set of location information of an aggregate set of location information.” As discussed for limitation [10.pre], Skagerwall discloses a system including directory servers (“a system having a plurality of location servers”) for storing and retrieving application server addresses and application identifiers associated with an object tag (collectively “location information”). (See Section IX.A.1.i.) Skagerwall also discloses that each directory server (“each of the plurality of location servers”) stores “a unique set of location information of an aggregate set of location information” for the reasons discussed for limitation [10.a]. (See Section IX.A.1.ii.)

103. Skagerwall does not explicitly disclose “a method of scaling at least one of capacity and transaction rate capability in a location server” in the system. A person of ordinary skill in the art would have been motivated to implement this feature in view of Vingralek.

104. As discussed for limitation [10.b], Vingralek and Skagerwall are in the same field of art, and a person of ordinary skill in the art would have been motivated to consider the teachings of Vingralek when implementing Skagerwall’s system/method. (*See* Section IX.A.1.iii.) As discussed therein, Vingralek discloses a system where a file is organized into buckets spread across servers. (*See* Section IX.A.1.iii; Ex. 1007, 4.) Vingralek employs a local load control at each server, preventing server overloading by redistributing buckets when a specified server utilization is reached. (Ex. 1007, 6.) Vingralek also implements an “explicit mapping of buckets onto servers,” allowing “additional flexibility to redistribute the load of an overloaded server” by, e.g., splitting bucket(s) of the server, migrating a bucket to a newly-acquired server, or migrating a bucket to an existing server. (*Id.*, 6-7.) Vingralek determines the utilization/performance capacity of the existing servers and decide on splitting versus migrating buckets, and acquiring new servers versus redistributing across existing servers. (*Id.*, 7 (“The decisions about splitting versus migrating a bucket and acquiring a new

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

105. Vingralek describes two load capacity concepts: the “feasible load capacity,” i.e., “the maximum number of keys that the server can keep without being overloaded,” and the “panic load capacity,” i.e., “the number of keys after which a server is no longer capable of accepting additional keys.” (*Id.*, 10.)

Vingralek describes several load management techniques for splitting/migrating buckets, and for moving buckets to an existing server or acquiring a new server. (*Id.*, 11-14.) For example, if a server exceeds its feasible load capacity, the file advisor begins monitoring that server’s capacity (*id.*, 12) and determines whether to split/migrate buckets, or to acquire a new server, based on factors such as the number of keys added to the server since the overload message was sent, as well as on whether other metrics such as global utilization have exceeded specified thresholds (*id.*). If a server ultimately reaches the panic load capacity, then the file

advisor identifies a second server having enough unused performance capacity to accept some buckets from the first at-capacity server, and sends a migrate message to the first server, triggering the transfer of buckets from the first server to another (second) server. (*Id.*, 13.) Thus, Vingralek discloses scaling the capacity of a server by, e.g., increasing the effective capacity by transferring buckets of information to another server when certain thresholds (e.g., load capacity concepts) are reached.

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

107. A person of ordinary skill in the art would have been motivated to include scalability features similar to those disclosed by Vingralek when implementing a system similar to that of Skagerwall to manage the directory

servers' load capacities while minimizing the number of servers storing location data, so as to reduce the cost of scaling up the distributed system. For example, when a directory server is no longer able to accept additional requests to add object tags (e.g., reaching panic load capacity as discussed in Vingralek), consistent with Vingralek's teaching, a person of ordinary skill in the art would have been motivated to move/transfer certain object tag(s) and associated object data, including application server addresses and application identifiers (both "associated locations"), to a second/another directory server to allow the system of directory servers to continue fulfilling requests. Additionally, a person of ordinary skill in the art would have found it beneficial to implement Vingralek's teaching as it strives for the objective of cost-conscious scalability by not acquiring new servers unless the load on existing servers exceeds certain thresholds. (*Id.*, 10-13.)

108. Moreover, a person of ordinary skill in the art would have found Vingralek's method beneficial because while Skagerwall discloses the database of directory servers can be scaled arbitrarily (Ex. 1006, 4:12-15, 8:15-19), it does not explicitly disclose the specific processes for scaling that system of servers, e.g., how or when a server's load capacity is exceeded, which Vingralek teaches. A person of ordinary skill in the art would have been motivated to include features similar to Vingralek's method of scaling server capacity when implementing

Skagerwall's distributed system of directory servers, such that the combined system includes a method of scaling the capacity of a directory server in the distributed system of directory servers (“[a] method of scaling at least one of capacity and transaction rate capability in a location server in a system having a plurality of location servers”).

109. A person of ordinary skill in the art would have had a reasonable expectation of success combining the teachings of Skagerwall and Vingralek. As discussed above, both references are in the same field of art. (*See above.*) Moreover, Skagerwall discloses that its directory servers may be scaled arbitrarily. (Ex. 1006, 4:12-15, 8:15-19) Thus, combining the teachings of Skagerwall and Vingralek would have involved modifying the Skagerwall system in a manner that Skagerwall suggests is feasible—implementing a means of scaling the distributed database of directory servers. Additionally, the combination of Skagerwall and Vingralek would have involved applications of known technologies (e.g., known distributed systems (Skagerwall)) according to known methods (e.g., known methods of scaling and managing load capacity of a distributed system (Vingralek)) to yield the predictable result of a system implementing a method of scaling the distributed database of directory servers, as discussed above.

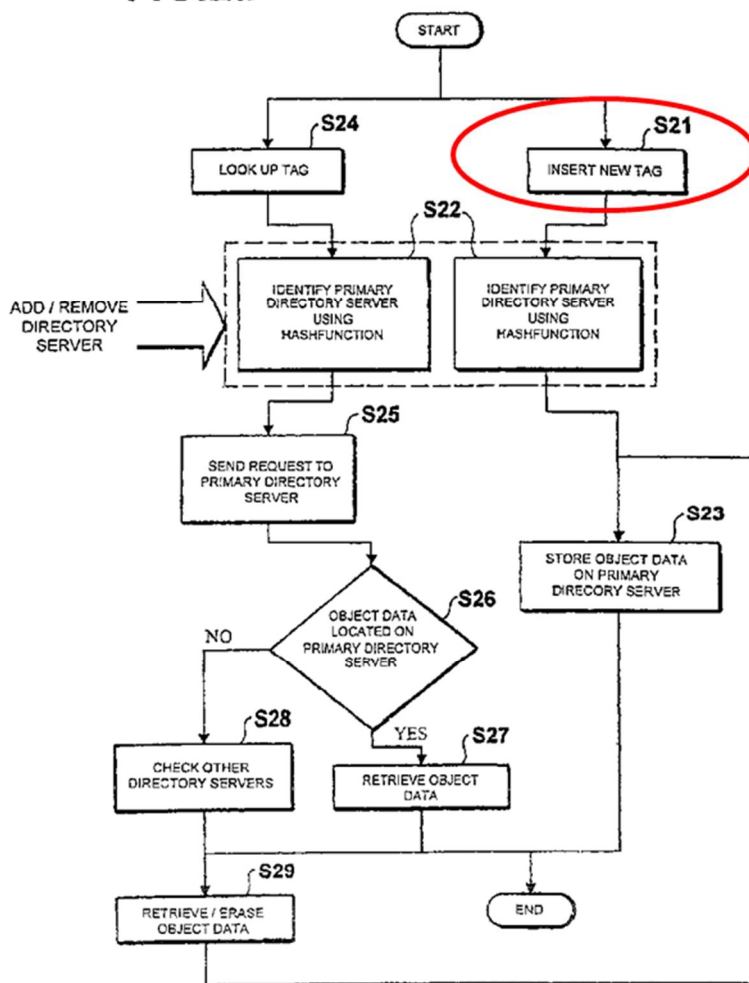
Accordingly, Skagerwall in view of Vingralek discloses and/or suggests this limitation.

- ii) **[17.a] “providing a transfer protocol configured to transport identifier and location information, the location information specifying the location of information related to the identifier;”**
- iii) **[17.b] “storing location information formatted according to the transfer protocol at a first location server;”**
- iv) **[17.c] “receiving an identifier and a location relevant to the identifier at the first location server;”**

110. Skagerwall discloses and/or suggests these limitations. **First**, Skagerwall discloses and/or suggests “transport[ing] identifier and location information, the location information specifying the location of information related to the identifier,” “storing location information...at a first location server,” and “receiving an identifier and a location relevant to the identifier at the first location server.” As discussed for limitation [10.pre], Skagerwall discloses a network of directory servers (each a “location server”), where each server stores mappings of object tags (“identifier”), application server addresses, and application identifiers (the three types of information in the mapping collectively “location information”), where such a mapping specifies the location of application data packets (“information”) associated with an object tag (“location information specifying the location of information related to the identifier”). (*See* Section IX.A.1.i.)

111. With respect to Figure 2a, Skagerwall describes inserting new object data into the database of object data distributed across directory servers. (Ex. 1006, 12:3-4.) Skagerwall discloses that a request for insertion of new object data (including a mapping of object tag, application server addresses, and application identifiers (collectively “location information”)), corresponding to an object tag (“identifier”) that is included in the request, is received by the system kernel in step S21 (red) (“transport identifier and location information, the location information specifying the location of information related to the identifier”). (*Id.*, 12:5-7; *see also id.*, 14:29-56 (describing a similar “request...to insert object data” with respect to Figure 4).)

FIG.2a



(*Id.*, FIG. 2A.) In step S22, a target directory server (“first location server”) for storing the object data is identified by applying a hash function to the object tag (“identifier”) and using the result as an index for searching the hash table. (*Id.*, 12:8-21, 14:37-53 (describing similar identification of target directory server with respect to Figure 4).) In step S23, the request to store the object data (including the disclosed “location information”) associated with a specific object tag

(“identifier”) is sent to the target directory server (“transport identifier and location information, the location information specifying the location of information related to the identifier”; “receiving an identifier and a location relevant to the identifier at the first location server”), which performs steps for storing/inserting the object data (“storing location information...at a first location server”). (*Id.*, 12:22-26.)

112. I have been asked to consider a scenario in which Skagerwall does not explicitly disclose transporting both “identifier” and “location information” to a directory server as part of the request to insert new object data. In such a scenario, a person of ordinary skill in the art would have been motivated to implement Skagerwall’s system in this manner. As discussed for limitation [10.pre], the directory servers each store object tags, application server addresses, and application identifiers in the form of a set of mappings, which map object tags (“identifier”) to all associated object data (including all the application server addresses and application identifiers associated with a specific object tag). (*See* Section IX.A.1.i.) Thus, a person of ordinary skill in the art would have been motivated to modify Skagerwall’s system to implement the request to add new object data such that the request includes the object tag itself and any associated object data to be added. A person of ordinary skill in the art would have found it beneficial to implement this feature because aligning the insertion and storage of

object data would have allowed for simplified storage of the object tag/object data mappings, without requiring additional steps (beyond the insertion request) to associate the object tag and object data on the directory server. In such a modified system, when a directory server receives a request to add new application server addresses and application identifiers (both “location[s]”), that request would include the object tag (“identifier”) and the application server addresses and application identifiers themselves to update the set of mappings (“transport identifier and location information, the location information specifying the location of information related to the identifier; “receiving an identifier and a location relevant to the identifier at the first location server”).

113. A person of ordinary skill in the art would have also had a reasonable expectation of success in implementing this modification in the Skagerwall-Vingralek combination. Implementing features similar to those disclosed in Skagerwall would have involved modifying the Skagerwall-Vingralek combination in a manner that Skagerwall suggests is feasible—aligning the processes for insertion and storage of new object data. Indeed, Skagerwall discloses storing the object data on the directory servers as part of the insertion process (Ex. 1006, 12:1-26), and also discloses storing the object data alongside the associated object tag (*id.*, 8:43-63, 15:15-20, 15:49-52). Moreover, such a combination would have

involved applications of known technologies (e.g., known systems for managing data and associated location information (Skagerwall)) according to known methods (e.g., known methods of inserting and storing location information (Skagerwall)) to yield the predictable result of a communication system in which object data insertion requests include both the object tag and associated object data.

114. **Second**, Skagerwall discloses and/or suggests “**providing a transfer protocol configured to transport identifier and location information**, the location information specifying the location of information related to the identifier.” Because Skagerwall discloses the access server constitutes the link between the user and system kernel (Ex. 1006, 16:39-40), a person of ordinary skill in the art would have understood that the above-described request to insert object location information would have been sent from the access server to a directory server. Thus, for reasons similar to as discussed for limitation [10.a], a person of ordinary skill in the art would have formatted the insertion request, like the information request discussed for limitation [10.a], according to known protocols, such as HTTP (“providing a transfer protocol configured to transport identifier and location information”), given that Skagerwall discloses that its system may be used with the Internet and requests may be made via HTTP. (See Section IX.A.1.ii; Ex. 1006, 9:59-62, 18:2-7, 18:24-26.)

115. **Third**, Skagerwall discloses “storing location information **formatted according to the transfer protocol** at a first location server.” As discussed above and for limitation [10.a], Skagerwall discloses and/or suggests formatting communications between access server and directory server (e.g., communications including object data) according to a transfer protocol such as HTTP. (See Section IX.A.1.ii.) I have been asked to consider a scenario in which Skagerwall does not disclose *storing* object data according to a transfer protocol. In such a scenario, a person of ordinary skill in the art would have been motivated to store the disclosed “location information,” at least temporarily, according to the format the information is transmitted in, i.e., HTTP standards, at the directory server (“storing location information formatted according to the transfer protocol at a first location server”). Moreover, as discussed for limitation [10.a], a person of ordinary skill in the art would have been motivated to retrieve the “location information” in the HTTP format/standards. (See Section IX.A.1.ii.) Thus, a person of ordinary skill in the art would have also found it beneficial to store such information according to the format that it is transmitted in.

116. A person of ordinary skill in the art would have also had a reasonable expectation of success in implementing this modification in the Skagerwall-Vingralek combination. Implementing features similar to those disclosed in

Skagerwall would have involved modifying the Skagerwall-Vingralek combination in a manner that Skagerwall suggests is feasible—implementing the system in line with a transfer protocol. (Ex. 1006, 9:59-62, 18:2-7, 18:24-26.) Moreover, such a combination would have involved applications of known technologies (e.g., known communication systems (Skagerwall)) according to known methods (e.g., known transfer protocols (Skagerwall)) to yield the predictable result of a communication system in which object data is stored at a directory server according to known transfer protocols (i.e., HTTP).

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

117. Skagerwall discloses and/or suggests this limitation. As discussed for claim 16, Skagerwall discloses that each directory server (“data location server”)⁸ has a memory means (“location store”) that stores object tags (“comprising a plurality of identifiers”) and the application server addresses and application

⁸ Skagerwall discloses a “data location server” under the Interpretation from Other Proceedings for “location server” for the same reasons as discussed for limitation [10.pre]. (Sections VIII, IX.A.1.ii.)

identifiers (both “location[s]”) associated with each object tag (“each identifier associated with at least one location”). (*See* Section IX.A.6.) A person of ordinary skill in the art would have thus understood that when a target directory server (“first data location server”) stores the received object tag (“the received identifier”) and associated application server addresses and application identifiers (both “the received location”), the directory server would store the information in the memory means (“location store”) such that the object tag is associated with the corresponding location information (“storing the received location in a location store at the first data location server...wherein the received location is associated with the received identifier in the location store”).

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

118. The Skagerwall-Vingralek combination discloses and/or suggests this limitation. As discussed for limitation [17.pre], in the Skagerwall-Vingralek combination, if a directory server (“first data location server”) reaches the panic load capacity (“when a performance criterion of the first location server reaches a predetermined performance limit”), a person of ordinary skill in the art would have been motivated to move/transfer certain object tag(s) and associated object data to a second/another directory server to allow the system of directory servers to

continue fulfilling requests. (Section IX.A.7.i.) As discussed for limitation [10.pre], the object data includes associated application server addresses and application identifiers (both “locations”). (See Section IX.A.1.i.) Thus, the Skagerwall-Vingralek combination discloses and/or suggests transferring a portion of the object tags (“identifiers”) and associated application server addresses and application identifiers (both “locations”) to the second directory server (“transferring a portion of the identifiers and associated locations to a second data location server”) when the panic load capacity is reached (“when a performance criterion of the first location server reaches a predetermined performance limit”).

8. Claim 23: “The method of claim 17, wherein the performance criterion comprises an amount of available persistent storage space in the first location server.”

119. The Skagerwall-Vingralek combination discloses and/or suggests this limitation. As discussed for limitations [17.pre] and [17.e], the panic load capacity (“performance criterion”) is “the number of keys after which a server is no longer capable of accepting additional keys.” (Ex. 1007, 10, 21 (“[T]he panic load capacity of a server is expressed in terms of the maximum number of keys after which the server would start thrashing.”); Sections IX.A.7.i and IX.A.7.vi.) While Vingralek expresses this capacity as a number of keys (similar to Skagerwall’s object tags, i.e., identifiers) that a server may hold, Vingralek explains that it

assumes that the key size is a fixed/constant amount of storage space. (Ex. 1007, 21 (“[T]he record size and the key size were fixed throughout the simulation.”), 22 (displaying simulation parameter settings in which the panic load capacity is 11000 records, record size is 10 Kbytes, and key size is 100 Bytes).) Thus, a person of ordinary skill in the art would have recognized that in the Skagerwall-Vingralek combination the panic load capacity factors in and/or represents the amount of persistent storage space available on a directory server (“the performance criterion comprises an amount of available persistent storage space in the first location server”).

9. **Claim 30: “The method of claim 17, wherein transferring a portion of the identifiers and associated locations to a second location server when a performance criterion of the first data location server reaches a predetermined performance limit further comprises monitoring the performance criterion and automatically transferring the portion of identifiers and associated locations when the first location server reaches the predetermined limit.”**

120. The Skagerwall-Vingralek combination discloses and/or suggests this limitation. As discussed for limitation [17.pre], Vingralek discloses that after a server reaches the feasible load capacity, the file advisor begins monitoring that server to determine if it hits other thresholds, e.g., the panic load capacity (“monitoring the performance criterion”). (See Section IX.A.7.i.) When a server hits the panic load capacity (“when the first location server reaches the

predetermined limit”), the file advisor begins identifying a second server for transferring buckets/information to, and sending a migrate message to the first server to trigger transferring of the information (“automatically transferring” information “when the first location server reaches the predetermined limit”). (*Id.*) For reasons similar to those discussed for limitation [17.pre], a person of ordinary skill in the art would have been motivated to implement in the Skagerwall-Vingralek combination processes similar to those described above, to monitor the directory server’s load capacity and automatically transfer the object tags (“the portion of identifiers”) and associated object data (e.g., application server addresses and application identifiers (both “associated locations”)) when the monitored server reaches the panic load capacity (“when the first location server reaches the predetermined limit”).

B. Skagerwall in View of Vingralek and Krasner Discloses and/or Suggests Claim 11

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

121. Skagerwall in view of Vingralek and Krasner discloses and/or suggests this limitation. Skagerwall discloses in addition to storing mappings of object tags, application server addresses, and application identifiers (collectively “location information”), the directory servers may more broadly store object data

“associating each of the objects with at least one of the application data packets.”

(Ex. 1006, 4:13-32.) While the Skagerwall-Vingralek combination does not expressly disclose that the directory servers may store geographic location information, a person of ordinary skill in the art would have been motivated to implement this feature in view of Krasner.

122. Like Skagerwall, Krasner relates to a communication system in a client-server architecture. Krasner discloses a wireless communication system including location servers that store location information pertaining to a mobile GPS unit. (Ex. 1009, Abstract (“A method and an apparatus for operating a satellite positioning system (SPS) receiver in a client/server architecture.”), 4:10-15 (“The system...includes a location server 25 which communicates through a wireless communication system with the communication system which is coupled to or part of the mobile GPS unit 12. The basestation 25 typically includes storage 26 for storing a time sequence of differential GPS and satellite ephemeris information.”).) Thus, Krasner is similar to Skagerwall, which discloses a wireless communication system containing a network of directory servers that store location information pertaining to various objects. (Ex. 1006, 7:37-43 (“transmission of information...maybe performed via wireless communication”), 8:15-28.) A person

of ordinary skill in the art would have therefore been motivated to consider the teachings of Krasner when implementing Skagerwall's system/method.

123. Krasner additionally discloses a "system for tracking the location of a mobile GPS receiver over time as it moves." (Ex. 1009, 3:56-57; *see also id.*, 1:48-51 ("Recently, GPS receivers have been used with radio transmitters, such as a cellular telephone or a mobile telephone in a car to transmit the position of the receiver as it moves."), 2:39-41 ("[I]t is desirable to provide an improved method and system for providing multiple sets of position information over a period of time through a mobile GPS unit.")) The mobile GPS unit receives GPS signals at various positions and determines pseudoranges from those signals, which it stores along with a corresponding time stamp in its memory. (*Id.*, 5:49-53.) Krasner's system includes a location server (or a basestation server) in communication with the mobile GPS unit being tracked. (*Id.*, 4:10-13.) The mobile GPS unit transmits the pseudoranges and time stamps to the location server when certain event(s) occur, e.g., a predetermined period of time has lapsed since the last transmission or an emergency. (*Id.*, 4:66-5:24 ("When the predetermined type event does occur,...the stored pseudoranges and the corresponding time stamps are transmitted via a wireless communication system...to the location server."), 5:49-6:15, 6:54-7:23.) The location server uses the transmitted information to "provide

concierge services or routing information” to a user. (*Id.*, 7:23-28.) Thus, Krasner discloses a location server storing geographical location information (“the location information comprises geographic location information”) of a mobile GPS unit and providing such information to a user.

124. A person of ordinary skill in the art would have been motivated to include features similar to those disclosed by Krasner when implementing Skagerwall’s system. Skagerwall discloses a flexible system in which a directory server (“location server”) may generally store object data, including various types of location information (e.g., object tags, application server addresses, application identifiers) that provide the location of application data packets (i.e., data) associated with an object. (Ex. 1006, 8:15-28.) Further, Skagerwall discloses that when object data associated with a specific object tag is retrieved from a directory server, the object data may be directly transmitted to a user instead of, for example, using the object data to retrieve application data packets and sending those to the user. (*Id.*, 4:20-32, 7:49-52, 8:28-37, 12:54-62.) Thus, a person of ordinary skill in the art would have been motivated to consider modifying Skagerwall’s system to store different types of object data on directory servers, or modifying to enable certain applications in which object data (e.g., location information) is shared directly with a user. As discussed above, Krasner discloses location servers (akin

to Skagerwall's directory servers) storing geographical location information and providing that information directly to users. (*See* above.) A person of ordinary skill in the art would have recognized such a feature would have beneficially allowed a user to track the location of an object over time, while offering advantages over similar systems, as discussed above. For example, Skagerwall discloses that the objects may "be any products or items of the physical world." (Ex. 1006, 6:45-48 ("[t]he system is arranged to link the plurality of objects, which can...be any products or items of the physical world to various kinds of information related to the objects"), 9:49-50 ("As noted before, the application servers provide various services associated to physical objects.").) A person of ordinary skill in the art would have understood that it would have been beneficial to track the geographical locations of physical objects, e.g., products purchased or the vehicle carrying the purchased products, and allow a user to request the tracked geographic location information via a server, e.g., a directory server, using an object tag identifying the object (i.e., mobile GPS unit), similar to as described for limitation [10.pre]. (*See* Section IX.A.1.i.)

125. A person of ordinary skill in the art would have also had a reasonable expectation of success combining the Skagerwall-Vingralek combination with Krasner. As discussed above, the references are in the same field of art. (*See*

above.) Additionally, implementing the above-discussed features similar to those of Krasner would have involved modifying the Skagerwall-Vingralek combination in a manner that Skagerwall suggests is feasible—implementing the system such that it stores different types of object data on directory servers, and providing the same according to a request. Moreover, such a combination would have involved applications of known technologies (e.g., known communication systems (Skagerwall)) according to known methods (e.g., known methods of storing geographic location information and providing the same (Krasner)) to yield the predictable result of a communication system that stores geographic location information on directory servers and provides such information per request, as discussed above.

C. Skagerwall in View of Vingralek and Sato Discloses and/or Suggests Claims 24-26

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

126. Skagerwall in view of Vingralek and Sato discloses and/or suggests this limitation. As discussed for limitations [17.pre] and [17.e], the Skagerwall-Vingralek combination discloses and/or suggests a method of scaling capacity of a location server, and transferring certain identifiers and associated locations to a second data location server when a performance criterion of the first location

server reaches a predetermined performance limit. (Sections IX.A.7.i, IX.A.7.vi.)

While the Skagerwall-Vingralek combination does not disclose scaling a location server's capability where the performance criterion comprises a transaction rate limit, a person of ordinary skill in the art would have been motivated to implement this feature in view of Sato.

127. Like Skagerwall and Vingralek, Sato describes a “client-server system built on a distributed system” where the client “issue[s] a service request to a specific server and receive[s] a service.” (Ex. 1011, ¶[0003].) Sato's system uses another set of servers, i.e., service location servers, which provide location information of the specific server that provide services to clients. (*Id.*, ¶[0003] (“One method to efficiently investigate the existence of a server that provides the service requested by a client and to efficiently obtain information for establishing the communicating means with the server is to use a server that services location information of various servers (hereinafter referred to as a service location server).”).) Sato further discloses a server partitioning method that partitions the service range and manages location information in a distributed manner “when the load on the service location server exceeds a predetermined threshold.” (*Id.*, ¶[0005].) Thus, Sato is similar to Vingralek, which discloses a method of migrating information from one server to another when the first server reaches a

performance criterion threshold. (Ex. 1007, 5-8, 13.) A person of ordinary skill in the art would have therefore been motivated to consider the teachings of Sato when implementing the Skagerwall-Vingralek combination.

128. Sato additionally discloses that each “service location server” includes a “load measuring means” measuring “service load information” of the server (Ex. 1011, ¶[0009]), where the measured “service load information” includes “the maximum number of service executions per unit time...after starting the service of the server” (*id.*, ¶[0012]). Whether to partition the server “is determined by comparing the maximum number of service executions [per unit time]...with [a predetermined] threshold value” (*id.*, ¶¶[0013], [0029]), and if partitioning is necessary, then information on the server is migrated to another server (*id.*, ¶¶[0029]-[0030]). Thus, Sato discloses a method of scaling transaction rate capability in a server, where information is transferred from a first to second server when the maximum number of service executions per unit time (or execution rate) reaches a predetermined threshold value (“performance criterion comprises a transaction rate limit”).

129. A person of ordinary skill in the art would have been motivated to include features similar to those disclosed by Sato when implementing the Skagerwall-Vingralek combination. For example, Vingralek assumes that the

“server instruction rate” (i.e., server processing speed) is constant. (Ex. 1007, 22 (noting that the server instruction rate is held at a constant 10 MIPS (million instructions per second)).) While Vingralek generally describes a homogenous system (*id.*, 21), its approach is “applicable also to heterogeneous servers where servers...may have different load capacities” (*id.*, 26). Thus, a person of ordinary skill in the art would have been motivated to modify the Skagerwall-Vingralek system to account for heterogeneous servers where the server instruction rate (processing speed) may vary based on the characteristics of each server. Sato’s method of partitioning a server based on a maximum service execution rate is advantageous because it allows for dynamic distributions of server workload upon reaching a predetermined threshold, rather than requiring distribution in advance, thereby “increasing the flexibility of the system.” (Ex. 1011, ¶[0073].)

Additionally, Sato’s method would have been beneficial because it would have allowed implementing a partitioning method in a system of heterogeneous servers, where processing speed may vary. Consistent with the above and in such a combined system, for example, each directory server would be configured to have a set maximum service execution rate threshold such that, in addition to transferring identifier/location mappings to a second directory server based on the load capacity principles discussed in Vingralek (e.g., panic load capacity), it would

also transfer identifier/location mappings to a second directory server when a first server's maximum service execution rate exceeds a predetermined threshold.

130. A person of ordinary skill in the art would have also had a reasonable expectation of success combining the teachings of Skagerwall, Vingralek, and Sato. As discussed above, these references are in the same field of art. (*See* above.) Moreover, it would have involved modifying the Skagerwall-Vingralek system in a manner that Vingralek suggests is feasible—implementing methods of partitioning servers in a heterogeneous system. Additionally, the combination of Skagerwall, Vingralek, and Sato would have involved applications of known technologies (e.g., known system for partitioning servers (Vingralek)) according to known methods (e.g., known methods of partitioning when a maximum service execution rate exceeds a threshold (Sato)) to yield the predictable result of a system in which directory servers are configured to transfer information to other servers when a transaction rate limit is reached, as discussed above.

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

131. The Skagerwall-Vingralek-Sato combination discloses and/or suggests this limitation. As discussed for claim 24, the performance criterion is a maximum service *execution* rate (“transaction rate limit”). (*See* Section IX.C.1.) A person of ordinary skill in the art would have understood that the maximum

service execution rate reflects the maximum speed at which a processor of a directory server operates (“transaction rate limit comprises a processor speed limit”). (Ex. 1006, 15:1-7 (disclosing that the directory server includes “a central processing unit CPU for controlling operations of the directory server.”).)

3. Claim 26: “The method of claim 17, wherein the transaction rate limit comprises a network connection bandwidth limit.”

132. The Skagerwall-Vingralek-Sato combination discloses and/or suggests this limitation. Vingralek discloses the **performance** of the distributed network is “characterized by the network latency *NL* (i.e., the time required to process each packet, independently of the packet size) and the **bandwidth** *BW* (i.e., the transmission time that is needed to send one byte through the network).” (Ex. 1007, 22.) A person of ordinary skill in the art would have understood that the network bandwidth is of a finite value/number. Thus, Vingralek discloses “a network connection bandwidth limit.” A person of ordinary skill in the art would have understood that the disclosed “network connection bandwidth limit” is a factor of the “transaction rate capability” (recited in limitation [17.pre]) or “transaction rate limit” (recited in claim 26) of a directory server (“location server”) because a network’s bandwidth impacts the rate at which the servers may

receive requests from clients, and thus the maximum service execution rate of the servers. (*See* Section IX.A.7.i.)

133. With the above teachings, a person of ordinary skill in the art, when implementing the Skagerwall-Vingralek-Sato combination would have been motivated to “transfer[] a portion of the identifiers and associated locations to a second data location server when a performance criterion of the first location server reaches a predetermined performance limit” (as recited in limitation [17.e]), where the “performance criterion” is “the transaction rate limit compris[ing] a network connection bandwidth limit” (as recited in claim 26). (*See* Section IX.A.7.i.) For example, the bandwidth of the part of the network connected to a first directory server may be limited/reduced due to, e.g., available bandwidth being occupied by an increased number of client requests. This is particularly true as the servers may be heterogeneous having different load capacities as disclosed in Vingralek. (Ex. 1007, 26; Section IX.C.1.) As such, a person of ordinary skill in the art would have found it beneficial to conduct a process similar to as described for limitation [17.e], e.g., identifying another directory server that has sufficient network connection bandwidth to accept buckets from a first directory server and transferring the same. (Section IX.A.7.vi.) The process would have reduced the load of the first directory server, and increased the utilization of the

second directory server, consistent with the goal of Vingralek that “strives for the objective of ‘cost-conscious’ scalability, by aiming to minimize the number of servers that are used for providing the necessary performance capacity.” (Ex. 1007, 6.)

134. A person of ordinary skill in the art would have also had a reasonable expectation of success in implementing the above-discussed feature in the Skagerwall-Vingralek-Sato combination. Indeed, it would have involved modifying the Skagerwall-Vingralek-Sato system in a manner that Vingralek suggests is feasible—implementing methods of partitioning servers in a heterogeneous system. Additionally, the modification would have involved applications of known technologies (e.g., known system for partitioning servers and known network bandwidth limit (Vingralek)) according to known methods (e.g., known methods of partitioning when a maximum service execution rate exceeds a threshold (Sato)) to yield the predictable result of a system where directory servers are configured to transfer information to other servers when a network connection bandwidth limit is reached, as discussed above.

X. CONCLUSION

135. 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