Trials@uspto.gov
Tel: 571-272-7822

Paper 73
Entered: February 19, 2014

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

MICROSOFT CORPORATION
Petitioner

v.

PROXYCONN, INC.
Patent Owner

Case IPR2012-00026
Case IPR2013-00109
Patent 6,757,717

Before SALLY C. MEDLEY, THOMAS L. GIANNETTI, and MITCHELL G. WEATHERLY *Administrative Patent Judges.*

WEATHERLY, *Administrative Patent Judge*.

FINAL WRITTEN DECISION
35 U.S.C. § 318(a) and 37 C.F.R. § 42.73

I. BACKGROUND

A. Introduction

On September 18, 2012, Microsoft Corporation ("Microsoft"), filed a petition under 35 U.S.C. §§ 311-319 for *inter partes* review of claims 1, 3, 10–12, 14, and 22–24 of U.S. Patent No. 6,757,717 ("the '717 Patent"). IPR2012-00026, Paper 6 ("the '026 Petition"). We granted the '026 Petition as to certain challenges to the patentability of claims 1, 3, 10, and 22–24, and denied the '026 Petition as to all challenges to the patentability of claims 11, 12, and 14 on December 21, 2012. IPR2012-00026, Paper 17 ("the '026 Decision").

Soon afterward, on January 11, 2013, Microsoft filed a second petition for *inter partes* review, this time challenging the patentability of claims 6, 7, 9, 11, 12, and 14 of the '717 Patent. IPR2013-00109, Paper 1 ("the '109 Petition"). Microsoft concurrently filed a motion to join IPR2013-00109 with IPR2012-00026. IPR2013-00109, Paper 7. We granted the '109 Petition as to certain challenges to patentability of claims 6, 7, 9, 11, 12, and 14 of the '717 Patent. IPR2013-00109, Paper 14 ("the '109 Decision"). We also granted Microsoft's motion for joinder and joined IPR2013-00109 with IPR2012-00026. IPR2013-00109, Paper 15.

After institution and joinder of both trials, Proxyconn, Inc. ("Proxyconn") filed its Corrected Patent Owner's Response ("Resp."). Paper 45.[1] Proxyconn also filed Patent Owner's Corrected Motion to Amend ("Mot. Amend") in which Proxyconn moved to substitute claims 35–

[1] This reference to "Paper" and all other references to "Paper" from this point forward in this Final Written Decision are to papers filed in the joined proceeding, which is captioned as IPR2012-00026 and IPR2013-00109.

41 for claims 1, 3, 6, 10, 11, 22, and 23, respectively, if the Board were to cancel any of those challenged claims as unpatentable. Paper 44.[2] This Final Written Decision addresses challenges to the patentability of claims 1, 3, 6, 7, 9–12, 14, and 22–24. Because claims 1, 3, 6, 10, 11, 22, and 23 are found unpatentable, this Decision also addresses the patentability of proposed substitute claims 35–41.

B. The '717 Patent

The '717 Patent describes a system for data access in a packet switched network. Ex. 1002, Abstract. The system has a sender/computer including an operating unit, a first memory, a permanent storage memory, and a processor. The system also has a remote receiver/computer including an operating unit, a first memory, a permanent storage memory, and a processor. The sender/computer and receiver/computer communicate through the network. *Id*. The sender/computer further includes a device for calculating digital digests on data; the receiver/computer further includes a network cache memory and a device for calculating digital digests on data in the network cache memory; and the receiver/computer and/or the sender/computer includes a device for comparison between digital digests. *Id.*

As described in the '026 Petition, the '717 Patent provides a way to reduce the amount of redundant data transmitted over a network. '026 Petition, 4. The processes described in the '717 Patent check for the identity

[2] Proxyconn filed Patent Owner's Motion to Amend under 37 C.F.R. § 42.121 on May 21, 2013. Paper 37. In an Order entered June 20, 2013, Proxyconn was granted permission to file its Corrected Motion to Amend to address typographical errors and file corrected exhibits. Paper 43. Proxyconn filed its Corrected Motion to Amend later that same day.

between two sets of data by comparing respective digital fingerprints of that data. *Id.* As described in the Summary of the Invention:

> If a sender/computer in the network is required to send data to another receiver/computer, and the receiver/computer has data with the same digital digest as that of the data to be sent, it can be assumed with sufficient probability for most practical applications that the receiver/computer has data which is exactly the same as the data being sent. Then, the receiver/computer can use the data immediately without its actual transfer through the network. In the present invention, this idea is used in a variety of ways.

Ex. 1002, col. 2, ll. 16-24.

The patent discloses several embodiments. In one, a sender/computer required to send data to a receiver/computer initially sends a digital digest of the data. If the receiver/computer already has data with the same digital digest, it uses this data as if it were actually transmitted from the sender/computer. *Id*. at col. 2, ll. 26-31. This embodiment is illustrated in Figures 5-7. Figure 5 is reproduced below:

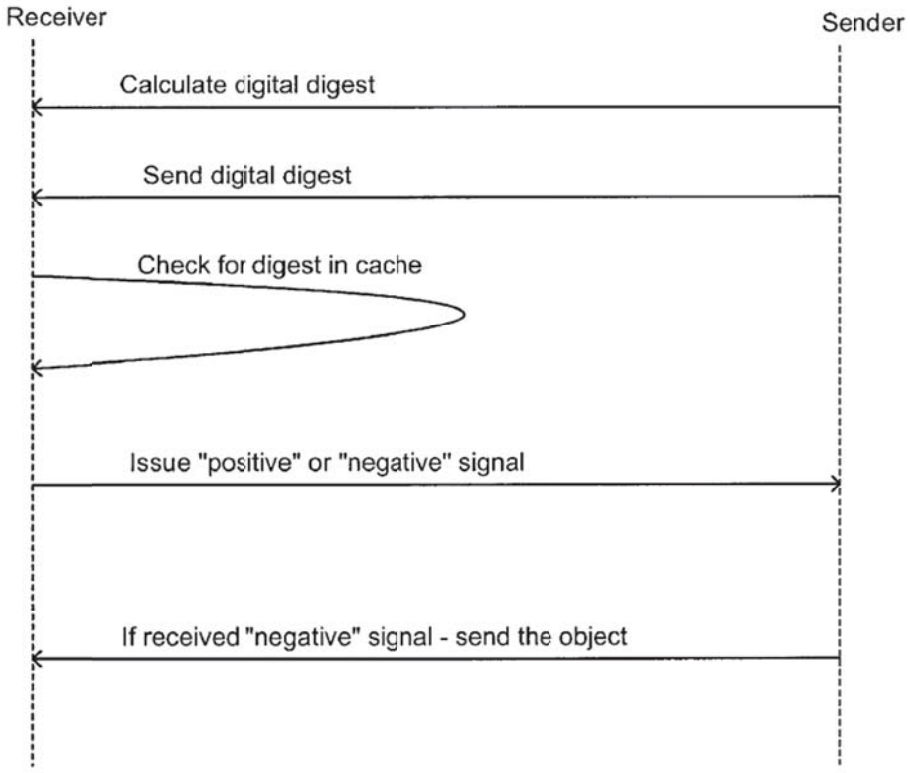


FIG. 5

Figure 5 is a schematic representation illustrating the interaction between a sender/computer and a receiver/computer according to the teachings of one embodiment of the '717 Patent. *Id*. at col. 5, ll. 49-51.

In this embodiment, the receiver/computer receives a digital digest from a sender/computer and searches its network cache memory for data with the same digest. If the receiver/computer finds such data, it uses that data as if the data were received from the sender/computer and issues a positive indication signal to the sender/computer. Otherwise it sends a negative indication signal to the sender/computer. *Id*. at col. 7, ll. 51–60.

In another embodiment, auxiliary digital digests for other data objects can be sent together with the principal digest. If the receiver/computer cannot find data having the principal digest, it searches for data with one of the auxiliary digests. If such data is found, the sender/computer is required

to send only the difference between the requested data object and the data object corresponding to the auxiliary digest. *Id*. at col. 2, ll. 31-37. The expression in the Specification "difference between the first data or data object and the second data or data object" means any bit sequence that enables the restoration of the first data, given the second data, the bit sequence, and the method employed in calculating the difference. *Id*. at col. 2, ll. 38–42. This embodiment is illustrated in Figures 8-10. Figure 8 is reproduced below:

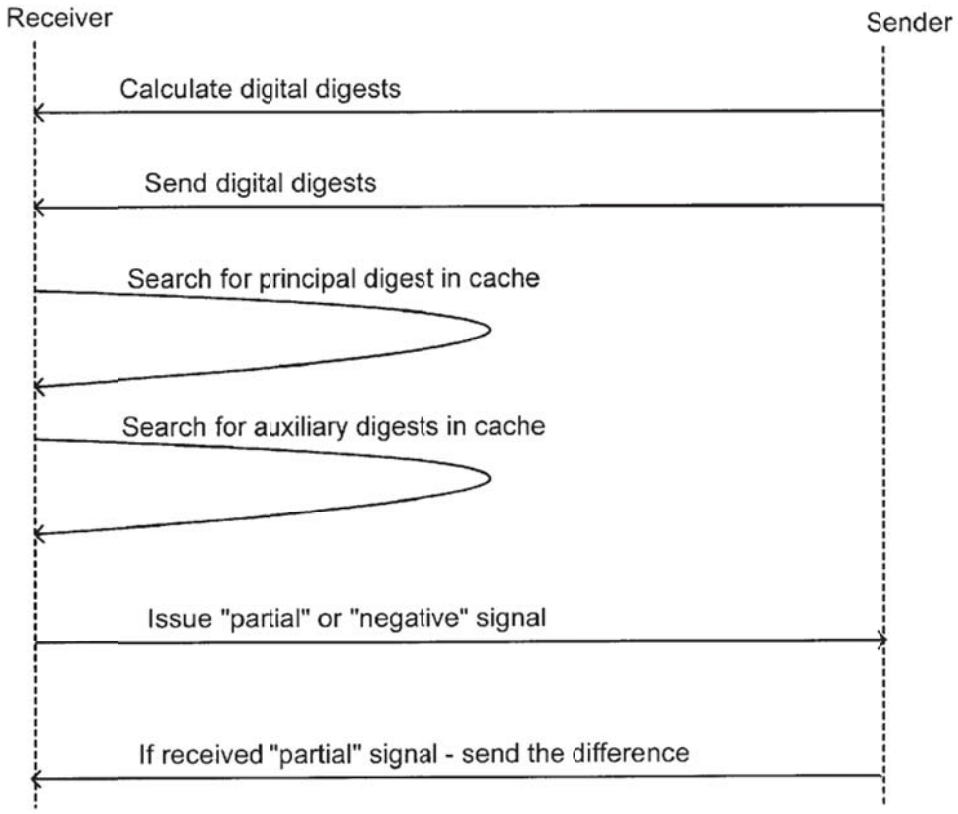


FIG. 8

Figure 8 is a schematic representation illustrating the interaction between a sender/computer and a receiver/computer according to the teachings of another embodiment of the invention. *Id*. at col. 5, ll. 59-61.

In this embodiment the sender/computer sends the principal and auxiliary (e.g., of a previous version of the data requested) digests to the receiver/computer. Upon receiving a message with these digital digests from the sender/computer, the receiver/computer searches its network cache memory for data having the same principal digest. If such data is found, the receiver/computer uses the data as if the data were received from the sender/computer and issues a positive indication signal to the sender/computer. Otherwise, the receiver/computer searches its network cache memory for data with the auxiliary digests. If it finds data with a digital digest substantially equal to one of the auxiliary digests, it issues a partial indication signal to the sender/computer, along with a reference to the digest. Otherwise it issues a negative indication signal to the sender/computer. *Id*. at col. 8, ll. 11-39.

C. Exemplary Claims

Claims 1, 6, 10, 11, and 22 are the independent claims among the challenged claims of the '717 Patent. Claims 1, 6, and 10 are directed to systems, and claims 11 and 22 are directed to methods. The independent challenged claims, which are illustrative of the claims at issue in this *inter partes* review, recite:

> 1. A system for data access in a packet-switched network, comprising:
>
> a sender/computer including an operating unit, a first memory, a permanent storage memory and a processor and a remote receiver/computer including an operating unit, a first memory, a permanent storage memory and a processor, said sender/computer and said receiver/computer communicating through said network;

said sender/computer further including means for creating digital digests on data;

said receiver/computer further including a network cache memory and means for creating digital digests on data in said network cache memory; and

said receiver/computer including means for comparison between digital digests.

6. A system for data access in a packet-switched network, comprising:

a gateway including an operating unit, a memory and a processor connected to said packet-switched network in such a way that network packets sent between at least two other computers pass through it;

a caching computer connected to said gateway through a fast local network, wherein said caching computer includes an operating unit, a first memory, a permanent storage memory and a processor;

said caching computer further including a network cache memory in its permanent storage memory, means for calculating a digital digest and means for comparison between a digital digest on data in its network cache memory and a digital digest received from said packet-switched network through said gateway.

10. A system for data access in a packet-switched network, comprising:

a sender/computer including an operating unit, a first memory, a permanent storage memory and a processor and a remote receiver/computer including an operating unit, a first memory, a permanent storage memory and a processor, said sender/computer and said receiver/computer communicating through a network;

said sender/computer further including means for creating digital digests on data, and

said receiver/computer further including a network cache memory, means for storing a digital digest received from said network in its permanent storage memory and means for comparison between digital digests.

11. A method performed by a sender/computer in a packet-switched network for increasing data access, said sender/computer including an operating unit, a first memory, a permanent storage memory and a processor and said sender/computer being operative to transmit data to a receiver/computer, the method comprising the steps of:

creating and transmitting a digital digest of said data from said sender/computer to said receiver/computer;

receiving a response signal from said receiver/computer at said sender/computer, said response signal containing a positive, partial or negative indication signal for said digital digest, and

if a negative indication signal is received, transmitting said data from said sender/computer to said receiver/computer.

22. A method for increased data access performed by a receiver/computer in a packet-switched network, said receiver/computer including an operating unit, a first memory, a permanent storage memory, a processor and a network cache memory, said method comprising the steps of:

receiving a message containing a digital digest from said network;

searching for data with the same digital digest in said network cache memory,

if data having the same digital digest as the digital digest received is not uncovered, forming a negative indication signal and transmitting it back through said network; and

creating a digital digest for data received from said network cache memory.

Ex. 1002, col. 10, l. 31 to col. 12, l. 45.

D. Remaining Challenges to the Patentability of Claims

We instituted this *inter partes* review in connection with the following challenges to the patentability of claims in the '717 Patent:[3]

1. Anticipation by Perlman: claims 1, 3, and 22-24;
2. Anticipation by Yohe: claims 1, 3, 6, 7, 10, 22, and 23;
3. Anticipation by Santos: claims 1, 3, 10, 22, and 23;
4. Anticipation by DRP: claims 6, 7, 9, 11, 12, and 14;
5. Obviousness over the combination of Perlman and Yohe: claims 1, 3, 10, and 22-24; and
6. Obviousness over the combination of Mattis and DRP: claims 6, 7, 9, 11, 12, and 14.

'026 Decision 25–26; '109 Decision 20.

II. ANALYSIS

A. Claim Interpretation

We interpret patent claim language in an *inter partes* review by ascribing to that language its broadest reasonable meaning in light of the specification of the patent. 37 C.F.R. § 42.100(b); Office Patent Trial

[3] The challenges to patentability are based upon five prior art references: US 5,742,820, issued Apr. 21, 1998 (Ex. 1003) ("Perlman"); US 5,835,943, issued Nov. 10, 1998 (Ex. 1005) ("Yohe"); Santos and Wetherall, INCREASING EFFECTIVE LINK BANDWIDTH BY SUPPRESSING REPLICATED DATA (June 1998) (Ex. 1004) ("Santos"); THE HTTP DISTRIBUTION AND REPLICATION PROTOCOL, W3C Note (August 25, 1997), retrieved from http://www.www3.org/TR/NOTE-drp-19970825 (IPR2013-00109, Ex. 1003) ("DRP"); US 6,292,880 B1, issued Sep. 18, 2001 (IPR2013-00109, Ex. 1004) ("Mattis").

Practice Guide, 77 Fed. Reg. 48,756, 48,766 (Aug. 14, 2012). We also interpret claim language according to its ordinary and customary meaning to one of ordinary skill in the art in the context of the entire disclosure. *In re Translogic Tech., Inc*., 504 F.3d 1249, 1257 (Fed. Cir. 2007).

We expressly interpret below only those claim terms that require analysis to resolve arguments related to the patentability of the challenged claims. Except as otherwise stated, we interpret the remaining claim terms as set forth in the '026 Decision and the '109 Decision.

1. Data Access

Each contested claim recites "data access." Ex. 1002, col. 10, l. 31 (claims 1, 3), col. 10, l. 64 (claims 6, 7, and 9), col. 11, l. 20 (claim 10), col. 11, l. 35 (claims 11, 12, and 14), col. 12, l. 30 (claims 22–24). Proxyconn urges that "data access" means "obtaining data . . . on a remote computer on a network, in response to a request from a client." Resp. 11 (citing Ex. 1002, col. 1, ll. 18–26; *id. at* col. 7, ll. 65–67). In support, Proxyconn cites portions of the Specification of the '717 Patent that describe exemplary data transmission sessions in which a network client "requests" data from a server. The first cited portion describes such interactions between a client and server as "prior art." Ex. 1002, col. 1, ll. 18–26. The second cited portion states: "This transaction begins with a receiver/computer sending a request to the sender/computer." *Id*. at col. 7, ll. 65–67. The phrase "[t]his transaction" refers to the interaction between the receiver/computer and the sender computer depicted in Figures 5–7 of the '717 Patent. *Id*. at col. 7, ll. 51–67.

By contrast, Microsoft contends that "data access" means "data acquisition." Microsoft Corporation's Reply to Patent Owner's Corrected

Response ("MS Reply"), 2 (Paper 46). Microsoft dismisses the portions of the Specification that Proxyconn cites as neither mentioning "data access" nor narrowly defining "data access." *Id*. Microsoft argues that other portions of the Specification imply that the step of the receiver/computer requesting data is merely optional. *Id*. (citing Ex. 1002, 8:37–39). Proxyconn's expert, Dr. Konchitsky, testified that the Specification describes scenarios in which a sender transmits data to a receiver without a request from the receiver. *See* Ex. 1024, 69:1–24, 71:8–22 (describing the data communication method illustrated in Figure 8 of the '717 Patent). Microsoft also points out that claim 32, which is not challenged, explicitly recites a method in which a client sends a request for data to a server. Paper 72, Final Hearing Transcript 10:9-12, 79:22–80:9 ("Tr.").

Both parties' interpretations of "data access" are too narrow. Neither the challenged claims nor the Specification expressly limits "data access" to require a "request from the client" as proposed by Proxyconn. The claims merely recite "data access." Even though the Specification describes examples in which the client requests data from a server, the Specification does not require that the client request data in all described embodiments of the claimed systems and methods. For example, the Specification expressly describes an embodiment in which "a sender/computer required to send data to a receiver/computer . . . initially sends a digital digest of the data." Ex. 1002, col. 2, ll. 26–28. "[L]imitations are not to be read into the claims from the specification." *In re Van Geuns*, 988 F.2d 1181, 1184 (Fed. Cir. 1993) (citing *In re Zletz*, 893 F.2d 319, 321(Fed. Cir. 1989)). We decline to do so here.

Microsoft's position is similarly unsupported by the claims themselves or the Specification. None of the challenged independent claims affirmatively recites that the receiver/computer *acquires* data from the sender/computer. Microsoft cites no portion of the Specification, and we find no support for the proposition that the Specification equates "data access" with "data acquisition."

We determine that the plain meaning of "data access" is clear. Independent challenged claims 1, 6, 10, 11, and 22 recite "access" as a noun modified by "data." Ex. 1002, col. 10, l. 31 (claims 1, 3), col. 10, l. 64 (claims 6, 7, and 9), col. 11, l. 20 (claim 10), col. 11, l. 35 (claims 11, 12, and 14), col. 12, l. 30 (claims 22–24). "Access" plainly means the "freedom or ability to obtain or make use of." MERRIAM WEBSTER'S COLLEGIATE DICTIONARY 6 (10th ed. 1999). We conclude, therefore, that the claimed systems and methods recite "data access" to refer to the freedom or ability to obtain or use data. Although obtaining or acquiring data requires access to that data, access to the data need not involve acquisition of that data.

2. Permanent Storage Memory

Claims 1, 3, 6, 7, 9, 10, and 22–24 recite "permanent storage memory." Ex. 1002, col. 10, l. 31 – col. 13, l. 8. Proxyconn argues that "permanent storage memory" means non-volatile memory that can be used for writing and reading data and does not refer to read-only memory ("ROM"). Resp. 12. The Specification states "an example of a permanent storage memory may be a disk drive, a flash RAM or a bubble memory." Ex. 1002, col. 7, ll. 38–40. In support of its proffered definition of "permanent storage memory," Proxyconn also cites Yohe's statement that "'[p]ermanent storage memory,' as used herein, includes but is not limited

to, disk drive, flash RAM or bubble memory, for example." Resp. 12 (quoting Ex. 1005, col. 3, ll. 5–7).

Microsoft counters that "permanent storage memory" is not restricted to non-volatile memory that permits multiple write operations, but may also include storage that is write-once, read-many ("WORM") memory. MS Reply 2. Microsoft contends that a CD optical storage disc, a type of non-volatile WORM memory, would constitute permanent storage memory. *See id*. (citing Ex. 1024, 88:7–89:12). Thus, the dispute centers on whether "permanent storage memory" encompasses ROM and other types of WORM types of non-volatile memory.

The testimony of both experts persuades us that a skilled artisan would interpret "permanent storage memory" to cover non-volatile memory that supports multiple write operations. Dr. Long equated the "permanent storage" described in the '717 Patent with a "disk" or "flash" memory. Ex. 1026, 97:15–98:10. Dr. Konchitsky testified that a skilled artisan would have considered "permanent storage memory," which enables writing or storing of information, to differ from "permanent memory," which can only be read after being written one time "in factory." Ex. 2002 ¶ 21. The ability to write data many times to permanent storage memory is consistent with the way that "permanent storage memory" is used in the context of at least claim 6. Claim 6 recites a "caching computer further including a network cache memory in its permanent storage memory." The presence of cache memory, which is likely to be written many times, in the "permanent storage memory" implies a capability to write data many times to the claimed "permanent storage memory." Because claim terms are normally used consistently throughout the patent, the usage of a term in one claim may

illuminate the meaning of the same term in other claims. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1314 (Fed. Cir. 2005). Therefore, we interpret "permanent storage memory" to mean any non-volatile memory that supports multiple write operations.

3. Sender/Computer and Receiver/Computer

Challenged claims 1, 3, 10, 11, 12, 14, and 22–24 recite either a "sender/computer" or "receiver/computer" or both. Ex. 1002, col. 10, l. 31 – col. 13, l. 8. Previously, we interpreted "sender/computer" to mean a computer that sends data and "receiver/computer" to mean a computer that receives data. '026 Decision 14. We also concluded that each of these respective computers can encompass multiple devices including intermediaries. *Id*.

Proxyconn argues that our interpretation is "inconsistent with the '717 Patent, is not the broadest *reasonable* interpretation of the claim terms, and should be revised to exclude separate intermediate computers such as gateways, proxies, routers, and caching computers." Resp. 13. Proxyconn contends that the Specification consistently refers to the sender and receiver computers as separate devices.

Microsoft contends that we correctly interpreted the computers to encompass multiple devices including intermediate devices. The Specification represents the receiver and sender computers (46, 42 respectively) in decidedly schematic form, as shown in Figures 4, 11, and 14, reproduced below.

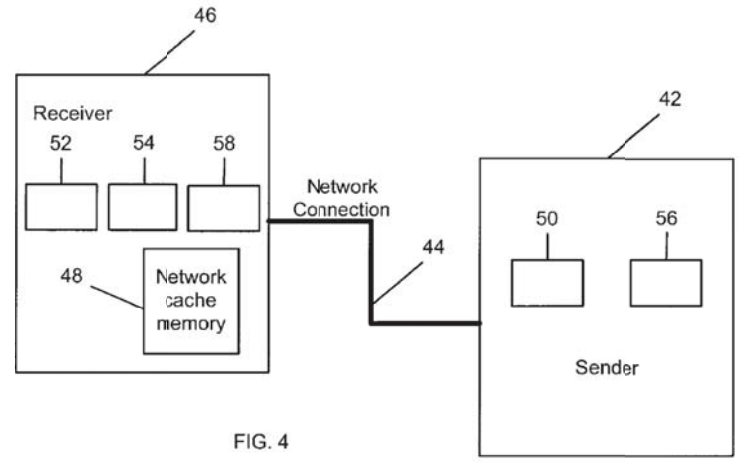


FIG. 4

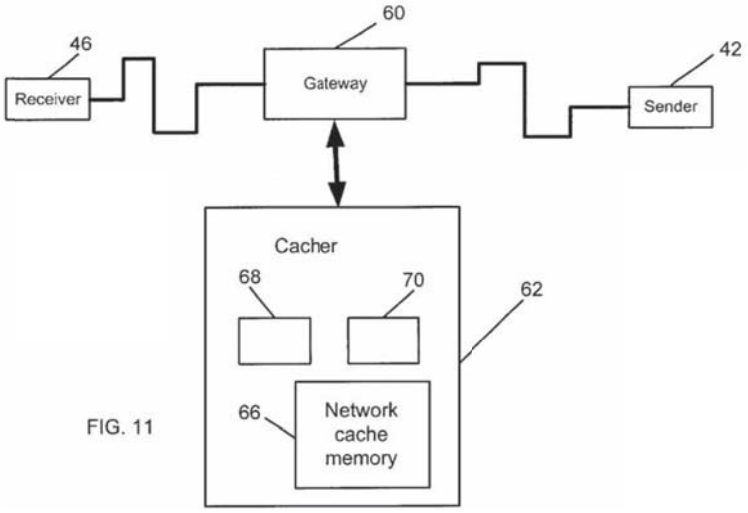


FIG. 11

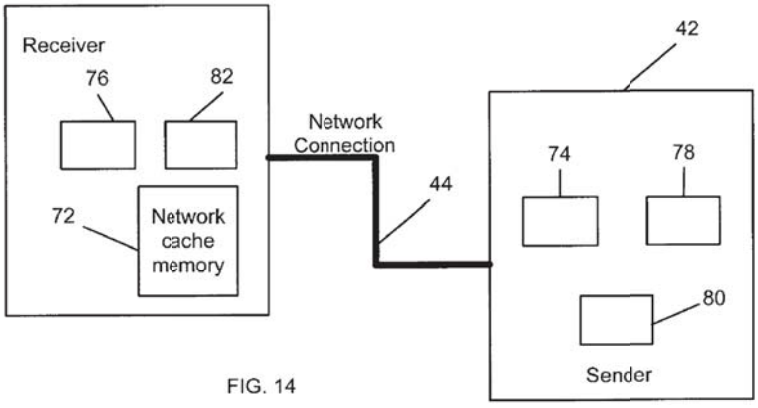


FIG. 14

Figures 4, 11, and 14, reproduced from top to bottom above, schematically illustrate the claimed receiver/computer and sender/computer in various network configurations as functional block diagrams.

Figure 4 illustrates receiver/computer 46 as a collection of functionally defined subsystems 48, 52, and 54, which are described as follows: "[T]he receiver/computer has calculating means 52 for calculating a digital digest on data stored in its network cache memory 48. The receiver/computer also has comparison means 54 for comparing between such a calculated digital digest and a digital digest received from the network." Ex. 1002, col. 7, ll. 32–37.

The Specification appears to have one instance in which a computer is described as being separate from or integral with another computer. The Specification implies that gateway 60 and caching computer 62 may be separate devices, but only by noting that "gateway computer **60** may be integrally formed with the caching computer." Ex. 1002, col. 9, ll. 6–8. The Specification, along with the above figures, conveys to a skilled artisan that the described computers, including the receiver/computer and the sender/computer, may or may not be located in separate housings. Accordingly, Proxyconn has not persuaded us to modify the original interpretation of "receiver/computer" and "sender/computer."

4. Gateway . . . Between at Least Two Other Computers

Independent claims 6 recites a "gateway . . . connected to said packet-switched network in such a way that network packets sent between at least two other computers pass through it." Ex. 1002, col. 10, l. 66 – col. 11, l. 2. Claims 7 and 9, which depend upon claim 6, also include the "gateway" and "two other computers." *Id*. at col. 11, ll. 13–20. Proxyconn asserts that "two other computers" refers to "the sender/computer and the receiver/computer." Resp. 15 (citing Ex. 1002, col. 2, ll. 44–47). The cited portion of the Specification, however, merely recites verbatim the language

of claim 6 relating to the gateway and two other computers. Therefore, the cited portion has not been shown to support Proxyconn's contention. Microsoft contends that no such limitation exists on the "two other computers" and that these computers may be *any* two other computers connected on the network to the gateway. *See* '109 Petition 13–14, Appendix A 5–6.

We agree with Microsoft. Claim 6 plainly and unambiguously recites "two other computers" as a limitation on the manner in which the "gateway" is "connected to said packet switched network." That is, the gateway is connected to the network so that "network packets sent between at least two other computers pass through it [i.e., the gateway]." Applying the broadest reasonable interpretation, we conclude that claim 6 does not limit which computers may constitute the "two other computers" between which the gateway is connected.

5. Means for comparison between digital digests

a. Claims 1 and 3

Claim 1 recites "means for comparison between digital digests." Resolution of the parties' arguments relating to whether Yohe anticipates claims 1, 3, and 10 requires that we interpret "digital digests" as recited in the comparison means. We interpret "digital digests" by reading claim 1 in its entirety. Claim 1 recites that both the sender and receiver include "means for creating digital digests on data." We conclude that the "digital digests" recited in the means for comparison refers to the "digital digests on data" that are recited earlier in claim 1 in the "means for creating."

b. Claim 10

Claim 10, like claim 1, recites that the receiver includes "means for comparison between digital digests." Also like claim 1, claim 10 recites that the sender includes a "means for creating digital digests on data." By contrast to claim 1, claim 10 does not recite a "means for creating" in the receiver. Ex. 1002, col. 11, ll. 20–33. Instead, the receiver includes a "means for storing *a digital digest* received from said network." *Id.* at col. 11, ll. 31–32 (emphasis added). The reference to "a digital digest" rather than "the digital digest on data" in the storing means implies that the receiver can store any type of digital digest received from the network.

Therefore, the "digests" that are compared in the "means for comparison" recited in claim 10 need not be the two digests on data created by the sender and receiver. Instead, the "means for comparison between digital digests" recited in claim 10 refers to structure that can compare any digital digest received from the network with any other digital digest.

6. Searching for Data with the Same Digital Digest

Claims 22–24 recite a step of "searching for data with the same digital digest." Proxyconn argues that the "searching" step requires the capability to identify particular data "with the same digital digest" from among a set of data that potentially contains multiple items. *See* Resp. 6, 20–21 (attempting to distinguish claims 22–24 from Perlman), 27–28 (attempting to distinguish claims 22 and 23 from Yohe), 35–36 (attempting to distinguish claim 23 from Santos). Microsoft contends that the '717 Patent equates "search" with "check for" and that the Specification never describes any "search" method other than "comparing two digest values for a match." MS Reply 4. Microsoft asserts that the recited step of "searching for data with the same

digital digest," merely requires comparing a digest for a data object received from the network with a digest of the receiver's copy of that data object. *Id*. at 5.

The Specification never expressly defines "search." Nonetheless, the plain meaning of "search" is: "to look into or over carefully or thoroughly in an effort to find or discover something." MERRIAM WEBSTER'S COLLEGIATE DICTIONARY 1053 (10th ed. 1999). Two dictionaries in the relevant field of computing technology define "search" as it would be understood by a skilled artisan as follows:

1. "To scan one or more data elements of a set in order to find elements that have a certain property," IBM DICTIONARY OF COMPUTING 600 (10th ed. 1993); and
2. "(information processing). To examine a set of items for those that have a desired property," IEEE STANDARD DICTIONARY OF ELECTRICAL AND ELECTRONICS TERMS 808 (3d ed. 1984).

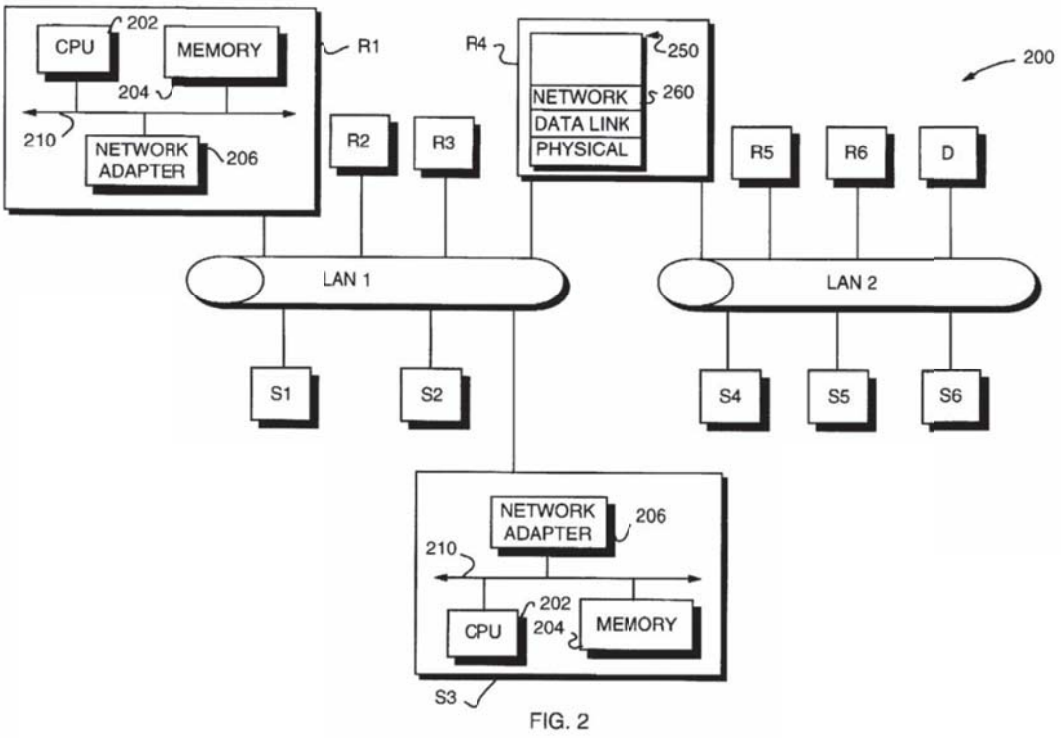
These dictionary definitions reflect that a skilled artisan would have understood "search" to involve analyzing a set of items to identify one particular item from among a set of items. A "set" refers to "a number of things of the same kind that belong or are used together," MERRIAM WEBSTER'S COLLEGIATE DICTIONARY 1071 (10th ed. 1999), or "[a] finite or infinite number of objects of any kind, of entities, or of concepts that have a given property or properties in common," IBM DICTIONARY OF COMPUTING 618 (10th ed. 1993). While a set can contain one item, a "search" for a desired member of a "set" requires a capability to examine more than one item to identify a particular item within that set. Therefore, we conclude that "searching for data with the same digital digest in said

network cache memory" requires an ability to identify a particular data object with the same digital digest from a set of potentially many data objects stored in the network cache memory.

B. The Prior Art

1. Perlman

Perlman generally relates to synchronization of information across a computer network. Ex. 1003, col. 1, ll. 6–8. Perlman's Figure 2 (reproduced below) is a block diagram of two computer networks to which multiple nodes, which include routers R1–R6, source nodes S1–S6, and a destination node D, are connected.



Perlman's Figure 2 is a block diagram of two computer networks to which multiple nodes are connected.

Perlman states that the "nodes are typically general-purpose computers" and that "[e]ach node typically comprises a . . . memory unit **204**" which may

include "storage locations typically composed of random access memory (RAM) devices." *Id*. at col. 5, ll. 40–47.

Packetized data is transmitted across the network with each packet having the address of its final destination and the address of the next node to which it will travel along the route to the final destination. *Id*. at col. 5, l. 65 – col. 6., l. 1. The final destination address remains constant, but the "next destination" address changes as the packet moves from node to node in the network. *Id*. at col. 6, ll. 1–4. Upon arrival of a packet to a router, the router determines the next destination address of the packet based on algorithms to calculate a path to the final destination. *Id*. at col. 6, ll. 5–24. For this mode of transmission to work, every router must determine and communicate its location in the network to other nodes on the network. *Id*. at col. 6, ll. 25–45. These network "maps" must be synchronized to ensure that data packets arrive at the correct final destination. *Id*. at col. 6, ll. 46–53.

Perlman synchronizes these network maps by having one designated router (e.g., R4) periodically calculate and send a digest of that map (called a "complete sequence numbers packet" or CSNP) to all other routers on the network. *Id*. at col. 6, l. 47 – col. 7, l. 55. When each of the other routers receives the CSNP digest from R4, each of those routers compares that received CSNP digest to a digest of the network topology calculated locally by the receiving router. *Id*. at col. 7, ll. 56–63.

Perlman also describes an "alternate embodiment" in which "high-level and low-level identifiers are bundled within the same hello message that is periodically broadcast by the designated router, **R4** to the other routers." *Id*. at col. 8, ll. 25–28. For this embodiment, the receiving router first compares its locally generated high-level digest with received high-

level digest 710. *Id*. at col. 8, ll. 32–35. If the two digests are not the same, then the receiving router calculates low-level digests for fragments of its database and compares each of these low-level digests with corresponding received low-level digests 725a–c. *Id*. at col. 8, ll. 36–39. Based on these comparisons, the receiving router determines which fragments of its database require updating. *Id*. at col. 8, ll. 39–42.

2. Yohe

Yohe generally describes an "apparatus for increased data access in a network [that] includes a file server computer having a permanent storage memory, [and] a cache verifying computer operably connected to the file server computer in a manner to form a network for rapidly transferring data." Ex. 1005, Abstract. Yohe's Figure 2, reproduced below, schematically illustrates the configuration of the data access network.

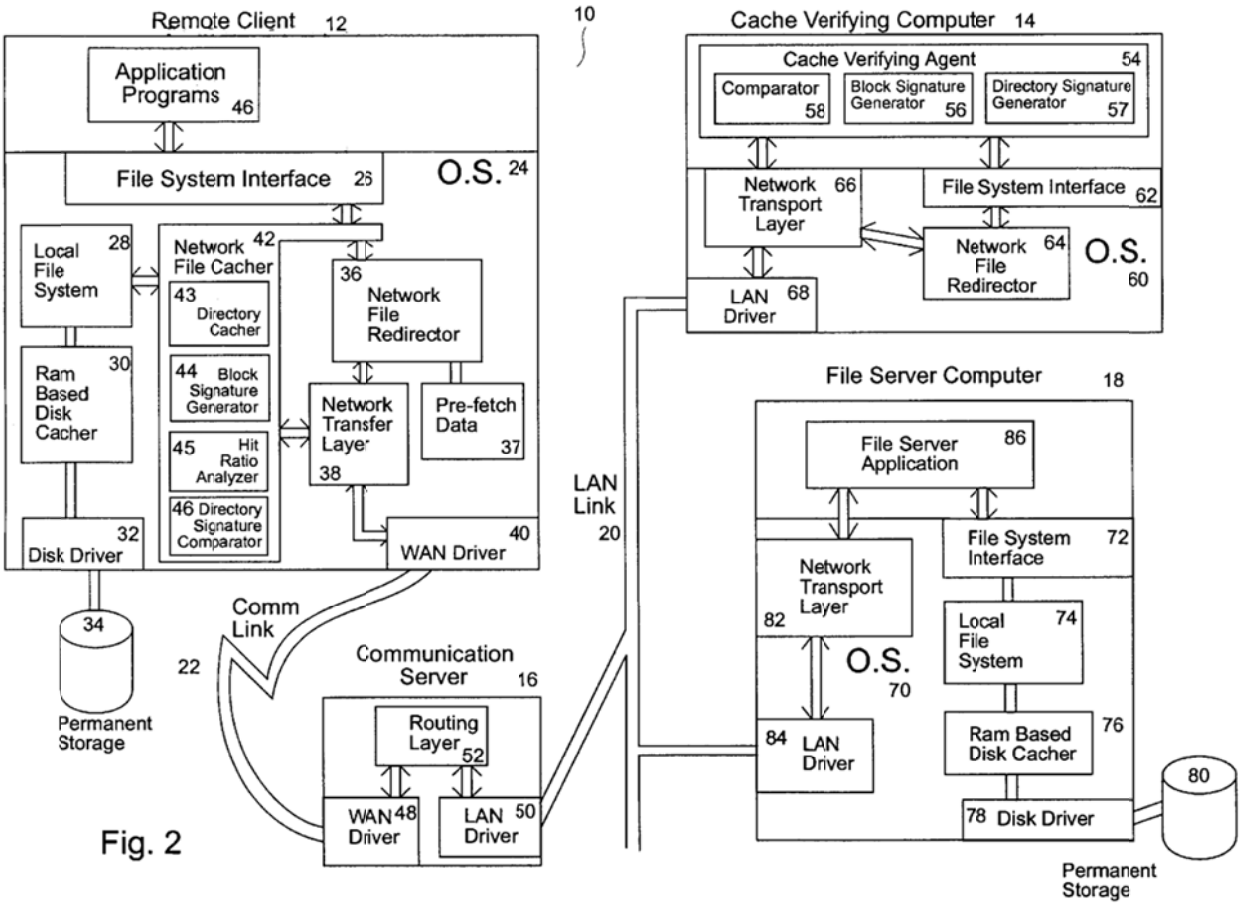


Fig. 2

Yohe's Figure 2 is a block diagram of a computer network including a remote client 12, cache verifying computer 14, communication server 16, and file server computer 18 that are connected to each other via either a LAN or WAN.

Yohe's apparatus reduces the time required for a remote client to access data on a file server using a caching computer and caching technique. *Id*. at col. 4, ll. 27–31.

Yohe's remote client computer 12 expressly includes a processor, operating system, permanent storage memory, and other memory. *Id*. at col. 2, ll. 51–54. The remote client computer also includes network file cacher 42 with block signature generator 44 and directory signature comparator 46. *Id*. at col. 5, ll. 1–6.

Similarly, Yohe's file server computer also expressly includes a processor, operating system, memory, and permanent storage memory. *Id*. at col. 3, ll. 22–24. The file server computer is "operably connected" to cache verifying computer 14, which includes "means for performing an operation on data stored in the permanent storage memory of the file server computer to produce a signature of the data characteristic of one of a file and directory." *Id*. at col. 2, ll. 43–51. Yohe describes two means for producing a signature, or digest, block signature generator 56 and directory signature generator 57. *Id*. at col. 5, ll. 14–17.

Yohe describes cache verifying computer 14 as having an operating system, a first memory, and a processor. *Id*. at col. 2, ll. 46–47. Cache verifying computer 14 incorporates cache verifying agent 54 consisting of block signature generator 56, directory signature generator 57, and comparator 58. *Id*. at col. 5, ll. 14–17. The cache verifying computer is also "operably connected" to the file server computer so that block signature generator 56 can create digests for data files stored in permanent storage 80. *Id*. at col. 2, ll. 43–51.

Yohe's client computer and the cache verifying computer thus each have capability to generate digests for data stored on the client and file server respectively. When Yohe's client computer requests data (e.g., to read a file stored in permanent storage on the file server), it generates a read request with an embedded digest and sends that request to the cache verifying computer. The cache verifying computer generates a digest for the requested file as that file exists in the permanent storage memory of the file server. The cache verifying computer then compares the two digests to determine whether the files stored on the remote client and file server are the

same. If not, the file server's version is sent to the remote client. If so, the remote client uses its locally stored version of the file. *Id*. at col. 6, ll. 22–37; figs. 6 and 7.

Yohe does not compare any particular received digest with more than one locally generated digest. Yohe describes two types of comparators for analyzing two versions of a digest. The first is comparator 58 in cache verifying computer 14, which compares digests for data files. *Id*. The second is directory signature comparator 46 in remote client 12, which compares one-by-one a series of digests for directory sub-objects that are received from the cache verifying computer with the locally generated digests for corresponding directory sub-objects. *Id.* at col. 7, l. 6 – col. 8, l. 25 (describing steps performed in DIRECTORY REQUEST function as shown in Figures 15 and 16).

3. Santos

Santos describes a compression architecture that prevents transmission of replicated data to increase bandwidth in a packet switched environment such as the Internet. Ex. 1004, 2. The bandwidth savings is achieved by transmitting repeated data as a short dictionary token, using caches of recently-seen data at both ends of the link to maintain the dictionary, and encode and decode the tokens. *Id.* at 5. The approach of Santos is based on the insight that the "fingerprint" of a data segment is an inexpensive name for the data itself, both in terms of space and time. *Id.* Santos uses the MD5 hash algorithm for his implementation, but states that other "fingerprints" could be used. *Id.* Figure 4 of Santos is reproduced below:

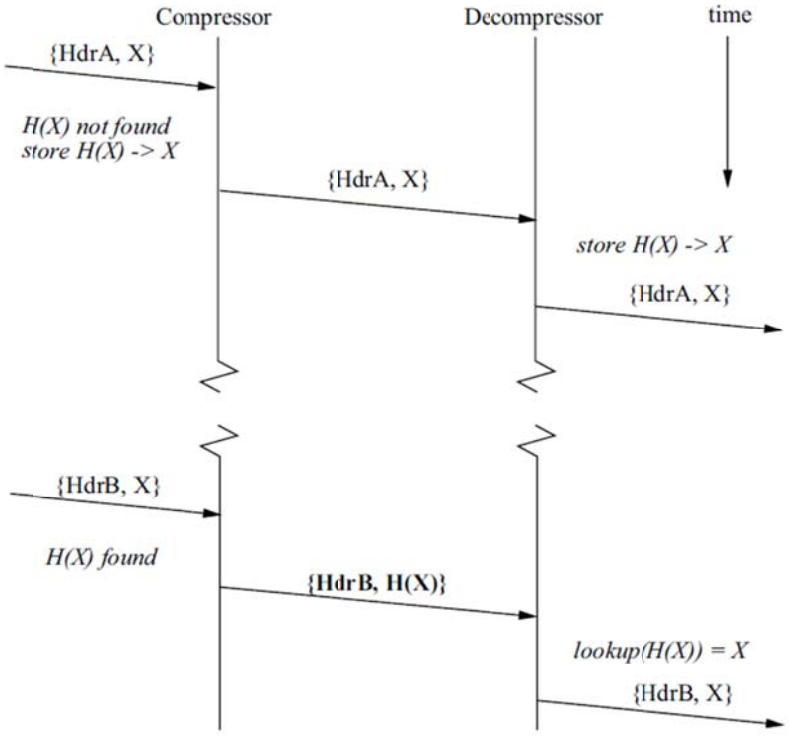


Figure 4: Compression protocol

Figure 4 of Santos shows a message exchange sequence from a sender (compressor) to a receiver (decompressor). Ex. 1004, 7.

The upper portion of the figure illustrates the sequence of events when the compressor receives a packet having header HdrA whose fingerprint H(X) is not in the cache. When this occurs, the compressor stores packet contents X in its cache, indexed by its fingerprint H(X), and forwards the header and contents across the link. The lower portion of the figure illustrates the sequence of events occurring when the compressor receives a packet having header HdrB and a fingerprint H(X) that is found in the cache. *Id*. at 7-8. When this occurs, the compressor sends the header and fingerprint, thus achieving a savings in bandwidth. *Id.* at 8. Santos implements the compressor and decompressor as two Pentium II based

machines with 128 MB of RAM and a cache of 200 MB running a Linux operating system. *Id*. at 9.

4. DRP

DRP describes a protocol for improving the efficiency and reliability of data distribution over HTTP. DRP, 2, ll. 12–13. One of the goals is to avoid downloading the same data more than once. *Id*. at 2, ll. 29–30. The protocol described in DRP makes use of content identifiers based on checksum technology. *Id*. at 2, ll. 37–39. A content identifier can be used uniquely to identify each piece of data or content and to determine whether two pieces of content are identical. An example of a checksum algorithm that can be used for this purpose is the MD5 message digest algorithm. *Id.* at 3, ll. 24–25.

To describe the exact state of a set of data files, DRP uses a data structure called an index. *Id.* at 4, l. 37. An index is a snapshot of the state of a set of files at a particular moment in time. It is typically stored in memory as a data tree structure, but to enable clients and servers to communicate this information over HTTP, an index can be described using XML. *Id*. at 4, ll. 39–42.

A DRP index is retrieved by giving a uniform resource locator ("URL") to the index. *Id*. at 5, l. 22. The index can be stored in any file and can be retrieved using a normal HTTP GET request. *Id*. at 5, ll. 22–23. Once the initial download is complete, a client can update content by downloading a new version of the index and comparing it against the previous versions of the index. Because each file entry in the index has a content identifier, the client can determine which files have changed and,

thus, determine the minimal set of files that need to be downloaded in order to bring the client up to date. *Id*. at 5, ll. 31–33.

An HTTP header field called Content-ID is used to specify the current correct version of the file that is requested by the client. The server can use the content identifier in the Content-ID field to determine if the requested version of the file can be delivered to the client. *Id*. at 7, ll. 30–32. If no content identifier is specified in the HTTP GET request, the server returns the current version of the file. *Id*. at 7, ll. 37–38. When a file is updated on the server, it will be downloaded by each of the clients that needs the new version. *Id*. at 8, ll. 3–4.

DRP notes that updates to files very often affect only small portions of the file, and it would therefore be much more efficient if the server could reply with only the parts of the file that have changed. *Id*. at 8, ll. 4–5. This is achieved using a "differential" GET request. When a file is modified, the client can issue a differential GET request for the file, which includes not only the content identifier of the desired version of the file, but also the content identifier of the current version of the file on the client. *Id*. at 8, ll. 11–13. In a differential GET request the content identifier of the file as it exists on the client is specified using the Differential–ID field in the HTTP header. *Id*. at 8, ll. 14–15. When the server receives a GET request that includes a Differential–ID field, it can look in its file cache for both versions of the requested file, using the content identifiers specified in the Content–ID field and the Differential–ID field. If both versions of the file are found, the server can compute the difference between the two files and return the difference, rather than the entire file. *Id*. at 8, ll. 25–28. If the server does not have access to the version of the file that is indicated by the differential

content identifier, it can ignore the differential content identifier and return the entire requested file. *Id*. at 8, ll. 32–33.

DRP also describes the use of proxy caching. In this application, an HTTP proxy can be made aware of the Content–ID and Differential–ID fields in HTTP requests and replies. *Id*. at 9, ll. 38–39. Because the content identifier is included in each GET request, the proxy can avoid accidentally returning the wrong version of the requested file. *Id*. at 9, ll. 39–40. The proxy can use the content identifier field to identify uniquely the content being transferred as the same content is likely to have the same content identifier, even when downloaded from multiple locations. The proxy can thus use this information to avoid multiple downloads. *Id*. at 9, ll. 43–45. The proxy can also use the Differential–ID header field to reply to differential GET requests. If both versions of the file are in the proxy's cache, the proxy can provide the differential reply. *Id*. at 10, ll. 1–2.

5. Mattis

According to Mattis, a key factor limiting the performance of the World Wide Web is the speed with which servers can supply information to clients via the Internet. Mattis, col. 1, ll. 53–55. Accordingly, client transaction time can be reduced by storing replicas of popular information objects in repositories geographically dispersed from the server. Each local repository for object replicas is generally referred to as a cache. *Id*. at col. 1, ll. 58–62. In some arrangements, the cache is located in a proxy server that is logically interposed between clients and the server. The proxy server is a "middleman gateway," acting as a server to the client, and the client to the server. *Id*. at col. 1. 66 to col. 2, l. 3. A proxy server equipped with a cache

is called a "caching proxy server," or just a "proxy cache." *Id*. at col. 2, ll. 3–5.

The proxy cache intercepts requests for resources that are directed from clients to the server. When the cache in the proxy has a replica of the requested resource that meet certain constraints, the proxy responds to the clients and serves the resources directly. *Id*. at col 2, ll. 6–11. In this arrangement, the number and volume of data transfers along the links are greatly reduced. As a result, network resources or objects are provided more rapidly to the clients. *Id*. at col. 2, ll. 11–14.

Mattis uses a "fingerprint" of the content that makes up the object itself to locate the object. *Id*. at col. 8, ll. 18–21. Specifically, the object key is a unique "fingerprint" or compressed representation of the contents of the object. A copy of the object is provided as an input to a hash function (e.g., MD5) and its output is the object key. Given a content fingerprint key, the content can easily be found in the cache. *Id*. at col. 8, ll. 23–36. In some embodiments of Mattis, for each of the objects, the cache also creates a name object key. The name key is created by applying a hash function to the name of the object. *Id*. at col. 8, ll. 55–58. Mattis recognizes that requests for objects typically identify requested objects by name, such as a URL, file system name, or network address. *Id*. at col. 9, l. 65 to col. 10, l. 4.

In one embodiment of Mattis, a lookup operation is used to determine whether a particular object identified by particular name is stored currently in the cache. *See id*. at fig. 9A. When the process is applied in the context of the World Wide Web, the name is a URL. *Id*. at col. 27, ll. 61–62. The cache converts the name of the object to a key value by passing the object

name or URL to hash function such as MD5. *Id*. at col. 27, ll. 63–67. The key is checked against a directory, and if the requested object is found, it is retrieved from storage and sent to the client. *Id*. at col. 28, ll. 10–14. If the object is not found in storage, the cache obtains a copy of the object from the appropriate server. *Id*. at col. 28, ll. 43–47.

C. Patentability of Original Claims

To prevail in its challenges to the patentability of claims, the petitioner must establish facts supporting its challenges by a preponderance of the evidence. 35 U.S.C. § 316(e); 37 C.F.R. § 42.1(d).

1. Anticipation

The Court of Appeals for the Federal Circuit summarized the analytical framework for determining whether prior art anticipates a claim as follows:

> If the claimed invention was "described in a printed publication" either before the date of invention, 35 U.S.C. § 102(a), or more than one year before the U.S. patent application was filed, 35 U.S.C. § 102(b), then that prior art anticipates the patent. Although § 102 refers to "the invention" generally, the anticipation inquiry proceeds on a claim-by-claim basis. *See Hakim v. Cannon Avent Group, PLC*, 479 F.3d 1313, 1319 (Fed. Cir. 2007). To anticipate a claim, a single prior art reference must expressly or inherently disclose each claim limitation. *Celeritas Techs., Ltd. v. Rockwell Int'l Corp.*, 150 F.3d 1354, 1361 (Fed. Cir. 1998). But disclosure of each element is not quite enough—this court has long held that "[a]nticipation requires the presence in a single prior art disclosure of all elements of a claimed invention *arranged as in the claim*." *Connell v. Sears, Roebuck & Co.*, 722 F.2d 1542, 1548 (Fed. Cir. 1983) (citing *Soundscriber Corp. v. United States*, 175 Ct.Cl. 644, 360 F.2d 954, 960 (1966) (emphasis added)).

Finisar Corp. v. DirecTV Grp., Inc., 523 F.3d 1323, 1334–35 (Fed. Cir. 2008). We must analyze prior art references as a skilled artisan would. *See Scripps Clinic & Res. Found. v. Genentech, Inc.*, 927 F.2d 1565, 1576 (Fed. Cir. 1991) (to anticipate, "[t]here must be no difference between the claimed invention and the reference disclosure, as viewed by a person of ordinary skill in the field of the invention").

For the reasons expressed below, Microsoft has demonstrated by a preponderance of evidence that Yohe anticipates claim 10; Santos anticipates claims 1, 3, 22, and 23; and DRP anticipates claims 6, 7, 9, 11, 12, and 14. Microsoft has failed to establish by a preponderance of evidence that Perlman and Yohe anticipate any of the remaining challenged claims.

a. Perlman

(1) Claims 1 and 3

Proxyconn argues that Perlman does not anticipate claims 1 and 3 because Perlman fails to disclose "permanent storage memory," but instead uses random access memory ("RAM"). Resp. 18 (citing Ex. 1003, col. 5, ll. 46–48)). Microsoft counters that Perlman describes permanent storage memory in the form of memory for storing an operating system. '026 Petition 17 (citing Ex. 1002, col. 5, ll. 41–52, fig. 2). We see no express statement in the cited portion of Perlman that its computers include any type of memory other than RAM. '026 Petition 17 (citing Ex. 1007, 12:5–17). Microsoft relies upon the Declaration of Darrell D. E. Long, Ph.D. Dr. Long opines, without explanation, that a skilled artisan would understand Perlman to disclose an "illustrative-embodiment router [that] is a 'general-purpose' computer . . . having, among other things, a hard disk or the like for storing

persistently data and application programs." Ex. 1007, 12:7–12 (citing Perlman, Ex. 1003, col. 5:41–52, fig. 2). However, Dr. Long also testified:

> Q. So your testimony is that each and every limitation of claim one is disclosed in Perlman, correct?
>
> A. We've got a computer — there may be — I'd have to go back and study Perlman. Maybe we don't mention permanent storage, whatever that means.
>
> Q. Okay.
>
> A. Okay. Although, certainly that's — I would consider that a triviality.
>
> Q. Okay.
>
> A. Okay. Digest, that's there. That's there. Comparisons are there. Maybe permanent memory is missing.

Ex. 1026, 172:10–22. When asked specifically how Perlman describes "permanent storage memory," Dr. Long responded, "I thought [Perlman] talked about flash in here. Certainly, it's something that a router can have and router[s] can and do have." Ex. 1026, 149:4–9. However, Dr. Long failed to identify how Perlman expressly discloses permanent storage memory. Based on his testimony, we conclude that a skilled artisan would understand Perlman to disclose routers that may or may not have permanent storage memory.

A finding of anticipation by inherency requires more than probabilities or possibilities. *Motorola Mobility LLC v. Int'l Trade Comm'n*, 737 F.3d 1345, 1350 (Fed. Cir. 2013). Based on the evidence discussed above, it is possible to infer that Perlman describes such permanent storage memory. However, Microsoft has not presented evidence that the computers or routers described by Perlman necessarily use permanent storage memory

as recited in claims 1 and 3. Therefore, Microsoft has failed to establish by a preponderance of evidence that Perlman describes a receiver/computer or sender/computer necessarily having "permanent storage memory." Without such evidence, we cannot conclude that Perlman anticipates claims 1 and 3.

(2) Claims 22–24

Proxyconn contends that Perlman fails to disclose the step recited in independent claim 22 of "searching for data with the same digital digest in said network cache memory." Resp. 20. Instead, Proxyconn urges that Perlman stores "one database identifier" that is merely "compared to the identifier received from the designated router." *Id.*; *see also* Ex. 2007, 13 (citing Ex. 1003, col. 8, ll. 32–42). Dr. Konchitsky testified, without citing any particular portion of Perlman, that Perlman's "receiving routers receive an identifier and each simply compares the received identifier with its existing identifier. The receiving routers are not searching for data files using the identifier as the key, or among multiple identifiers." Ex. 2002, ¶ 23.

Microsoft cites numerous passages from Perlman as meeting the recited "searching" step. '026 Petition, App'x. A, 17. All but one of the cited passages from Perlman describes the comparison of a received high-level database digest with a locally calculated high-level database identifier. The other cited passage (Ex. 1003, col. 8, ll. 22–49) describes an alternate embodiment in which received low-level digests are compared to locally generated low-level digests, if the received and locally generated high-level digests do not match. *See* part II.B.1 above (describing Perlman's use of high-level and low-level digests).

We conclude that Microsoft has failed to establish by a preponderance of evidence that Perlman meets the "searching" step recited in claim 22. Every portion of Perlman upon which Microsoft relies involves comparing one received digest to a corresponding locally calculated digest. These one-to-one comparisons cannot identify a particular data object from among a set of data objects as required by the "searching" step. Rather, the comparisons reveal whether a locally stored data object is synchronized with the corresponding remotely stored data object. Claims 23 and 24 depend from claim 22. We therefore, conclude that Perlman does not anticipate claims 22–24.

b. Yohe

(1) Claims 1, 3, and 10

Proxyconn first argues that Yohe fails to describe a sender/computer having permanent storage memory and means for creating digital digests of data. Resp. 21–23. Microsoft counters that the combination of Yohe's cache verifying computer 14 and file server 18 constitutes the claimed sender/computer. '026 Petition 16, App'x. A, 3; MS Reply 5–6; *see also* Ex. 1028, 16 (regarding annotated version of Yohe, Figure 2). We reject Proxyconn's argument because claims 1, 3, and 10 do not limit "sender/computer" to hardware residing in one housing. We have concluded that "sender/computer" may broadly encompass "multiple devices," and Proxyconn has directed us to no persuasive evidence that our interpretation is incorrect.

Proxyconn next argues that Yohe fails to describe a "receiver/computer" with means for creating digital digests on data. Resp. 23–24. Microsoft identifies Yohe's block signature generator 44 in

remote client computer 12 as the recited "means for creating digital digests on data." '026 Petition 20, App'x. A, 9; *see also* Ex. 1028, 13 (regarding annotated version of Yohe, Figure 2). We agree with Microsoft that Yohe's block signature generator 44 creates digests on data stored in Yohe's cache. Ex. 1005, col. 6, ll. 22–23.

Proxyconn also argues that Yohe fails to describe a "receiver/computer" with means for comparing received and locally generated digital digests on data. Resp. 24; *see also* Ex. 2007, 18–19 (regarding annotated version of Yohe, Figure 2). Proxyconn contends that comparator 58, which is part of cache verifying computer 14 and not remote client 12, compares the digests generated by block signature generators 44 and 56. Ex. 2007, 19 (annotating Yohe's Figure 2); *see also* Tr. 55:1–8 (identifying Ex. 1005, Yohe, col. 6, ll. 22 to col. 7, ll. 16 as confirming annotations on Yohe's Figure 2). Microsoft identifies Yohe's directory signature comparator 46 in remote client computer 12 as the recited "means for comparison between digital digests" that compares the digital digests on data that were separately created by the sender and receiver computers. '026 Petition 21, App'x. A, 10; *see also* Ex. 1028, 13 (regarding annotated version of Yohe, Figure 2).

We agree with Proxyconn that Yohe fails to describe a "receiver/computer including means for comparison of digital digests" as recited in claims 1 and 3. The two versions of digests on data that Yohe generates via block signature generators 44 and 56 are compared by comparator 58, which is not a component of remote client 12. For this reason, we conclude that Microsoft has not established by a preponderance of evidence that Yohe anticipates claims 1 and 3.

However, claim 10 does not require that the "means for comparison between digital digests" refers to comparing a digest generated by the receiver to a digest generated by the sender. *See* Part II.A.5.b above. Instead, we have concluded that the "means for comparison between digital digests" recited in claim 10 refers to structure that can compare *any* digital digest received from the network with another digital digest. Yohe's remote client 12 includes directory signature comparator 46 that compares a directory signature received from the cache verifying agent 54 (i.e., a portion of a sender/computer), with a directory signature retrieved from the remote client's cache. Ex. 1005, col. 8, ll. 5–11. Therefore, we agree with Microsoft that Yohe's directory signature comparator 46 is a "means for comparison between digital digests" as recited in claim 10. We determine that Microsoft has established by a preponderance of evidence that Yohe anticipates claim 10.

(2) Claims 6 and 7

Proxyconn argues that Yohe fails to describe a caching computer having network cache memory in its permanent storage memory as required in claims 6 and 7. Microsoft contends that permanent storage device 80, shown as part of Yohe's file server 18, constitutes the "permanent storage memory" of the claimed caching computer. '109 Petition 17, App'x. A,10. Essentially, Microsoft argues that the combination of Yohe's cache verifying computer 14 and file server 18 meet all the recited limitations for the caching computer of claim 6. Proxyconn contends that even if we were to agree that cache verifying computer 14, file server 18, and permanent storage device 80 were the claimed "caching computer," this combination still fails to describe network cache memory in the permanent storage

memory. Resp. 26–27. Microsoft fails to address this argument in its reply. *See* MS Reply 9. Our review of Yohe fails to identify any description of a network cache memory in permanent storage device 80. For this reason, we conclude that Microsoft has not established by a preponderance of evidence that Yohe anticipates claim 6 or its dependent claim 7.

(3) Claims 22 and 23

Proxyconn argues that Yohe fails to disclose the step recited in independent claim 22 of "searching for data with the same digital digest in said network cache memory." Resp. 27–28. Proxyconn contends that the only analysis of digests performed by Yohe's receiver (i.e., the remote client) is performed by Yohe's directory signature comparator 46, which compares one directory digest with a digest received from the cache verifying computer. *Id*. at 27. Proxyconn argues that such one-to-one comparisons are not "searching" as recited in claim 22. *Id*. at 27–28.

Microsoft does not dispute Proxyconn's characterization of the manner in which Yohe's directory signature comparator operates. MS Reply 11. Rather, Microsoft contends that Proxyconn's position is predicated on a flawed interpretation of "search" that requires more than a "single-comparison search." *Id*.

Based on our review of Yohe, the directory signature comparator 46 performs, at most, a series of one-to-one comparisons of received digests to locally generated digests for directory sub-objects. Ex. 1005, col. 7, l. 6 – col. 8, l. 25 (describing steps performed in DIRECTORY REQUEST function as shown in Figures 15 and 16). We conclude that such one-to-one comparisons do not identify a particular directory sub-object from among a set of directory sub-objects as required by the "searching" step. Rather, the

comparisons simply reveal whether a locally stored directory sub-object is synchronized with the corresponding remotely stored directory sub-object. Claim 23 depends from claim 22. We therefore conclude that Microsoft has not established by a preponderance of evidence that Yohe anticipates claims 22 and 23.

c. Santos

(1) Claims 1 and 3

Proxyconn raises two arguments allegedly distinguishing claims 1 and 3. We address each in turn below.

First, Proxyconn argues that Santos fails to describe a receiver/computer. Resp. 33–34. Proxyconn contends that Santos describes "a compressor and a decompressor, which are intermediate computers." *Id*. at 33. Microsoft counters that Proxyconn's argument rests upon an incorrect interpretation of "receiver/computer." We agree. We have interpreted "receiver/computer" to refer to "a computer that receives data." Santos's two computers on opposite ends of the communication channel can send and receive data. *See* Ex. 1004, 6 (stating that "[b]idirectional compression is achieved by using two instances of the protocol, one for each direction.") Thus, Santos uses "compressor" and "decompressor" to denote the function performed by each of two computers during transmission of specific data in a specific direction across the communication channel between them. Because data can move in both directions across that channel, both machines may function as a compressor (i.e., sender) or a decompressor (i.e., receiver). In the context of claims 1 and 3, the decompressor is acting as a "receiver/computer." We therefore reject Proxyconn's first argument.

Second, Proxyconn argues that Santos fails to describe "means for creating a digital digest on *data in the network cache memory*." Resp. 35. Proxyconn contends that because Santos's compressor and decompressor calculate the digest for data *after* the data is received but *before* the data is stored, the calculation of the digest is not on data in the network cache memory. *Id*. Proxyconn's argument rests upon an inferred interpretation of the creating means that requires a specific order of operations. More specifically, Proxyconn argues that "creating a digital digest on data in the network cache memory" implicitly requires that the receiver first read data from the network cache memory and then create a digest for that data.

Microsoft counters that claims 1 and 3 "do not require that the receiver first read the data from the cache and then and only then calculate a digest on it." MS Reply 8. Microsoft's argument also rests upon an inferred interpretation of the creating means. More specifically, Microsoft infers that the receiver's creating means need only be capable of creating a digest corresponding to data that is or will be stored in the network cache memory.

We conclude that Microsoft's interpretation is more consistent with the broadest reasonable interpretation of "means for creating digital digests on data in said network cache memory." Claim 1 and its dependent claim 3 recite systems, not methods, with hardware having recited functional capabilities. The functional language in claim 1 is not limited to a particular order of operations. "[A]pparatus claims cover what a device *is*, not what a device *does*. An invention need not *operate* differently than the prior art to be patentable, but need only *be* different." *Hewlett-Packard Co. v. Bausch & Lomb Inc.*, 909 F.2d 1464, 1468 (Fed. Cir. 1990); *see also Roberts v. Ryer*, 91 U.S. 150, 157 (1875) ("The inventor of a machine is entitled to the

benefit of all the uses to which it can be put, no matter whether he had conceived the idea of the use or not."). Here, the receiver/computer must be capable of creating a digest on data that is "in" the receiver's network cache memory.

We find that Santos describes a computer that can calculate a digest on data corresponding to data that will be stored in the receiver's network cache. When data not yet in Santos's compressor's cache is to be sent across the communication channel, Santos's compressor sends that data to the decompressor. Ex. 1004, 7. Santos's decompressor, upon receiving the data also creates a digest for that data and stores the digest and the data in the decompressor's cache. *Id*. Santos describes this case as follows:

> When the compressor receives a packet {HdrA, X} [i.e., data] to be forwarded over the link, where HdrA is the TCP/IP header and X is the data payload, it first computes H(X) [i.e., a digest], the fingerprint of X. If it finds that no entry indexed by H(X) [digest] exists in its cache, it stores X in its cache, indexed by H(X) [digest]. It then forwards the TCP/IP packet across the link. Upon receiving a TCP/IP packet forwarded over the channel, the decompressor also computes H(X) [digest], and stores X in its cache, indexed by H(X) [digest]. The TCP/IP packet is then output from the system.

Ex. 1004, 7, Figure 4.

Santos's decompressor thus calculates a digest for every data payload that it receives from the compressor and stores that data payload and its digest in its network cache memory. Ex. 1004, 7–8. We therefore reject Proxyconn's second argument and find that Santos describes a receiver/computer that includes "means for creating digital digests on data in said network cache memory." We determine that Microsoft has established by a preponderance of evidence that Santos anticipates claims 1 and 3.

(2) Claim 10

Proxyconn contends that Microsoft never challenged the patentability of claim 10 as anticipated by Santos and that our decision to institute a trial on this ground was "a mistake." Resp. 32–33, n.5. For this reason, Proxyconn never substantively addresses Santos as it relates to claim 10. *Id*. at 32–33. However, Microsoft's claim charts compared claim 10 to Santos. '026 Petition, App'x. A 11–13. That claim chart is part of Microsoft's petition, and we instituted review of claim 10 as anticipated by Santos on the grounds raised in the claim chart. Microsoft has thus proffered evidence to establish that Santos describes every limitation of claim 10. Proxyconn fails to rebut Microsoft's evidence. In the absence of countervailing evidence and argument, we determine that Microsoft has established by a preponderance of evidence that Santos anticipates claim 10.

(3) Claim 22

Claim 22 is directed to a method "performed by a receiver/computer." Ex. 1002, col. 12, ll. 30–31. Just as with claims 1 and 3, Proxyconn argues that Santos fails to describe a "receiver/computer." Resp. 33–34. For the same reasons described above in connection with claims 1 and 3, we find that Santos describes the claimed receiver/computer. Microsoft has proffered evidence to establish that Santos meets every other limitation of claim 22, and Proxyconn fails otherwise to rebut that evidence. Therefore, we determine that Microsoft has established by a preponderance of evidence that Santos anticipates claim 22.

(4) Claim 23

Proxyconn argues that Santos fails to describe "searching in predetermined locations of said permanent storage memory for data" for two reasons. Resp. 35–36. First, Proxyconn argues that Santos "lacks permanent storage memory," which is evident from the loss of "fingerprints H(X) and associated data" when Santos's computers are reset. *Id*. at 35 (citing "Ex. 1004 at § 3.2.1").[4] Proxyconn contends "[i]f either the data payload or fingerprints H(x) were stored in permanent storage memory, the data and fingerprint H(x) would be retained in memory, following either a reset or power cycle." Resp. 35–36.

In response, Microsoft characterizes Santos's compressor and decompressor as "general-purpose" computers that "necessarily" have ROM and a hard disk. MS Reply 11. Microsoft also contends that Santos' description of a 200 MB cache in a system with only 128 MB RAM implies that some cache must be in non-volatile memory. *Id*. (citing Ex. 1004, 7–9 and Figures 4 and 5). Microsoft does not directly address Proxyconn's contention that the loss of data in Santos's cache upon a reset demonstrates that Santos's cache does not reside in permanent storage memory.

Based on our own review of Santos, we reject Proxyconn's argument as not being supported by the evidence. Santos's only description of a reset relates to a "reset message" rather than a system reset or power interruption. Ex. 1004, 7 and 9. Santos describes using a reset message sent from the compressor to the decompressor or vice versa as a mechanism for handling a

[4] The cited portion of Santos does not address the effect of a "reset." Nonetheless, Santos describes a reset process elsewhere. Ex. 1004, 9 (in § 3.3).

lack of synchronization between the caches in each machine. *Id*. at 9. We understand these reset messages to be sent intentionally to reset the other cache to a known state in the event that the stored information about "illegal fingerprints" is lost (e.g., due to a compressor restart). *Id.* Santos does not describe where or how the compressor stores information relating to illegal fingerprints. Thus, we find Proxyconn's cited evidence to be inconclusive regarding the character of the cache memory.

Microsoft's evidence on the nature of the cache memory is more persuasive. Santos describes its compressor and decompressor in some detail as being "Intel-based Pentium II 300 MHz machines running Linux 2.0.31 with 128 MB of RAM." Ex. 1004, 9. Santos also states: "we limited the amount of memory available for the caches, . . . , to 200 MB each." *Id*. Santos also expressly describes repeated and numerous write operations to the cache. Ex. 1004, Figures 4 and 5 and accompanying text at 7–8.

We are persuaded that a skilled artisan would understand that Santos's 200 MB cache, which exceeds the available RAM in each machine, referred to a non-volatile memory that supports multiple write operations, which satisfies our interpretation of "permanent storage memory." We therefore reject Proxyconn's argument that Santos does not describe "permanent storage memory."

Second, Proxyconn argues that Santos fails to "search in predetermined locations" within the permanent storage memory. Resp. 36. Proxyconn asserts that "[s]earching in an index or files stored in memory is far different than searching at a predetermined location in memory, much less in predetermined locations in permanent storage memory." *Id*. The sole

support that Proxyconn provides for this assertion is the following testimony of Dr. Konchitsky.

> 51. I understand Santos to disclose that the fingerprints are not stored in permanent storage memory. Santos states that upon reset, such as through a power cycle or restart, the fingerprints H(x) and associated data is lost. [EX1004] at §3.3. This indicates to me, as it would to any person of ordinary skill in the art[,] that the fingerprints are not stored in permanent storage memory, and thus Santos could not logically teach searching permanent storage memory for fingerprints.

Ex. 2002 ¶ 51, Resp. 36. Dr. Konchitsky's testimony does not support Proxyconn's assertion that Santos searches "in predetermined locations" in memory. Rather, Dr. Konchitshy's testimony relates to whether Santos describes "permanent storage memory."

Microsoft responds that Santos "looks in at least two predetermined locations: it looks at H(X) and then, to identify any collisions, it looks at the stored payload associated with the H(X) and compares that to the argument payload." MS Reply 11 (citing Ex. 1004, 7–9, Figures 4 and 5). We determine that the evidence cited by Microsoft is persuasive and, therefore, reject Proxyconn's argument that Santos does not perform "searching in predetermined locations."

For the foregoing reasons, we conclude that Microsoft has established by a preponderance of evidence that Santos anticipates claim 23.

d. DRP

(1) Claims 6, 7, and 9

Proxyconn argues that DRP does not anticipate claims 6, 7, and 9 because it fails to describe the following three limitations on the caching computer recited in claim 6: (i) a "permanent storage memory;" (ii) "means

for comparison;" and (iii) "means for calculating a digital digest" as recited in claim 6. Resp. 37–38. Proxyconn argues that DRP also fails to describe additional limitations on the caching computer that are recited in dependent claims 7 and 9.[5] However, those arguments rely upon Proxyconn's contention that DRP fails to disclose a network cache in permanent storage memory. Resp. 38–39. Microsoft contends that DRP describes the claimed caching computer as any of its client and server computers, all of which have "permanent storage memory," "means for comparison," and "means for calculating a digital digest." MS Reply 8–9. Microsoft also contends that DRP's client and server computers meet each of the additional limitations on the caching computer that are recited in dependent claims 7 and 9.

Proxyconn's argument rests upon Proxyconn's interpretation of "two other computers" as excluding the "caching computer." We have rejected Proxyconn's interpretation as explained in Part II.A.4 above. Proxyconn does not otherwise dispute Microsoft's characterization of DRP's client or server computers as meeting all the limitations on the "caching computer" that are recited in claims 6, 7, and 9. Therefore, we determine that Microsoft has established by a preponderance of evidence that DRP anticipates claims 6, 7, and 9.

[5] Claim 7 depends upon claim 6 and further recites "said caching computer further includes means for calculating a digital digest for data in its network cache memory." Ex. 1002, col. 11, ll. 13–15. Claim 9 also depends upon claim 6 and further recites "said caching computer further includes means for storing said digital digest in said permanent storage memory." *Id*. at col. 11, ll. 18–20.

(2) Claims 11, 12, and 14

Microsoft identifies in great detail how DRP's clients and servers communicate according to the methods of claims 11, 12, and 14. '109 Petition, Appendix A 16–22; MS Reply 9–10 (citing DRP. 4:37–5:19, 5:22–32, 6:40–7:1, 7:20–29, 7:31–35, 7:37–39, 8:11–13, 8:29–31, 9:22–32; Konchitsky Tr. 91:18–94:7, 98:5–11, 108:11–109:18). Proxyconn argues that DRP fails to describe the step of "receiving a response signal from said receiver/computer at said sender/computer, said response signal containing a positive, partial or negative indication signal for said digital digest, and if a negative indication signal is received, transmitting said data from said sender/computer to said receiver/computer" as recited in independent claim 11. Resp. 40. Proxyconn contends that DRP's statement that the "client can use the index to automatically download the files that are specified" means that the client downloads specific files without ever sending a response signal to the server. *Id*. Proxyconn also argues that DRP fails to describe the requirement in claim 14 of "a response signal is sent containing a separate indication signal for each of said data objects." *Id*. at 41.

We reject Proxyconn's arguments regarding both claims. DRP describes that the client, after comparing the received digest of files with the digest for its cached versions, sends a GET request (when none of the digests for files match), a differential GET request (when some, but not all, of the digests for files in the cache match), or no request (when all digests for the files in the cache match). DRP, 5:22–32, 6:44, 7:20–28, 8:11–13. Dr. Konchitsky confirms the accuracy of Microsoft's position regarding the GET and differential GET requests that the client might send to the server. Konchitsky Tr. 108:11–109:18. The three types of responses that the client

sends to the server after receiving the index from the server and comparing it to the local index constitute the claimed "response signal containing a positive, partial or negative indication signal" of claim 11. These types of responses also correlate to the "separate indication signal" of claim 14. Proxyconn proffers no argument independently distinguishing claim 12 from DRP. Therefore, we conclude that DRP anticipates claims 11, 12, and 14.

2. Obviousness

"Section 103 [of 35 U.S.C.] forbids issuance of a patent when 'the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains.'" *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). To establish obviousness of a claimed invention, all the claim limitations must be taught or suggested by the prior art. *See CFMT, Inc. v. Yieldup Int'l Corp.*, 349 F.3d 1333, 1342 (Fed. Cir. 2003); *In re Royka*, 490 F.2d 981, 985 (CCPA 1974).

a. Yohe and Perlman

Microsoft contends that the combination of Yohe and Perlman renders claims 1, 3, 10, and 22–24 obvious under 35 U.S.C. § 103(a). Dr. Long testified that a skilled artisan would have considered Yohe and Perlman to be closely related technologies that are natural to combine because both references address the same problem and use the same algorithm in similar applications. Ex. 1007, 9:12–10:13. Dr. Long also testified that Perlman expressly suggests that its technology is suited for use in any type of node in a network for which synchronization of data is important. *Id*. at 11:1–7.

Proxyconn does not address the sufficiency of the combined teachings of Yohe and Perlman regarding the subject matter of claims 1, 3, 10, and 22–24. Instead, Proxyconn first argues that Perlman is not a proper reference to consider in an obviousness analysis because a skilled artisan would not consider Perlman to be analogous art to the claimed invention. Resp. 28–31. Dr. Konchitsky testified that the '717 Patent addresses increasing data access speed by conserving bandwidth while Perlman uses additional bandwidth to keep its nodes synchronized. Ex. 2002, 6. Dr. Konchitsky concluded that a skilled artisan would not consider Perlman to describe a viable way of increasing data access.

Dr. Long testified that Perlman, Yohe, and the '717 Patent all address the same problem: "the desire to reduce redundancy in network data transmissions where dynamic data previously sent over the network has been stored by the receiver for possible later reuse." Ex. 1007, 9:13–15. We find Dr. Long's statement of the problem addressed by the '717 Patent, Perlman, and Yohe to be persuasive. The '717 Patent states: "The performance gains realized by the present invention are derived from the fact that computers in common wide-area networks tend to repetitively transmit the same data over the network." Ex. 1002, col. 6, ll. 17–20. Both Perlman and Yohe describe reducing the use of bandwidth for data transmission as a way of improving network performance. Ex. 1007, 9:19–10:4 (citing Ex. 1003, col. 3, ll. 52–55; Ex. 1005, col. 4, ll. 32–40). Therefore, we reject Proxyconn's argument that a skilled artisan would not consider Perlman's teachings to be pertinent to the invention.

Proxyconn also argues that it would not have been obvious to incorporate Yohe's permanent storage memory into Perlman's router

because such memory would serve no function in Perlman's router. Resp. 31–32. "The test for obviousness is not whether the features of a secondary reference may be bodily incorporated into the structure of the primary reference. . . . Rather, the test is what the combined teachings of the references would have suggested to those of ordinary skill in the art." *In re Keller*, 642 F.2d 413, 425 (CCPA 1981) (citations omitted). Dr. Long testified at length about the reasons why a skilled artisan would have used permanent storage memory as taught by Yohe for the cache in Perlman. Ex. 1007, 12:4–15:9. Proxyconn proffers no persuasive evidence in support of its position. Therefore, we reject Proxyconn's argument that a skilled artisan would not have found it obvious to use permanent storage memory in the receiver/computer and sender/computer.

For the combination of Yohe and Perlman to render claims 1, 3, 10, and 22–24 obvious, the combination still must describe or suggest all limitations of the claims. Therefore, we analyze the teachings and the differences, if any, between the combination of Yohe and Perlman and the claims below.

(1) Claims 1, 3, and 10

As discussed in Part II.C.1 above, we determine that Yohe anticipates claim 10, but that neither Yohe nor Perlman anticipates claims 1 and 3. The only limitation recited in claims 1 and 3 not described by Yohe is the "means for comparison of digital digests on data" located in the receiver/computer. *See* Part II.C.1.b(1) above. However, Yohe describes comparator 58 in cache verifying computer 14 that would, if located in the remote client, meet the recited "means for comparison of digital digests." Additionally, Perlman describes performing the comparison of digital

digests on data in the receiver because Perlman's receiving router compares a received digest with a locally generated and stored digest.

Ex. 1003, col. 7, ll. 56–63.

Thus, we must decide whether a skilled artisan would have found it obvious to incorporate Perlman's comparing means into Yohe's remote client or to move or add Yohe's comparator 58 from cache verifying computer 14 into remote client 12. Both Perlman and Yohe describe devices that compare digital digests on data. Ex. 1003, col. 7, ll. 56–63; Ex. 1005, col. 6, ll. 22–37, Figures 6 and 7. Yohe suggests that "[i]t is recognized that other locations for the comparator [58] may exist." Ex. 1005, col. 13, ll. 32–34. Proxyconn has not suggested that Yohe's comparator would not be capable of performing the recited function of "comparison of digital digests on data." Incorporating Perlman's comparing means into Yohe's remote client or moving Yohe's comparator 58 into the remote client would have involved nothing more than ordinary skill and would have been using known devices to perform known functions to yield predictable results. We conclude that a skilled artisan would have found it obvious to include the recited "means for comparison of digital digests on data" in a receiver/computer. For these reasons, we conclude that Microsoft has established by a preponderance of evidence that the combination of Yohe and Perlman renders claims 1, 3, and 10 unpatentable as obvious under 35 U.S.C. § 103(a).

(2) Claims 22–24

As explained in Parts II.C.1.a(2) and II.C.1.b(3) above, neither Perlman nor Yohe describes, "searching" as required by claim 22. We cannot conclude that a claim would have been obvious when the prior art

does not describe every element recited in the claim. Claims 23 and 24 depend from claim 22. Therefore, we conclude that Microsoft has not established by a preponderance of evidence that claims 22–24 are unpatentable as obvious under 35 U.S.C. § 103(a).

b. Mattis and DRP

For the reasons expressed in Part II.C.1.d above, we determine that DRP anticipates claims 6, 7, 9, 11, 12, and 14. Because we cancel these claims based on DRP alone, we determine that Microsoft's challenge to the patentability of claims 6, 7, 9, 11, 12, and 14 as being obvious in light of Mattis combined with DRP is moot and do not address this challenge.

D. Proxyconn's Motion to Amend

Proxyconn moved to substitute claims 35–41[6] for challenged claims 1, 3, 6, 10, 11, 22, and 23, respectively, if the Board were to cancel any of those challenged claims as unpatentable. Mot. Amend 1. Proxyconn also requests that we enter claims 35–41 "in addition to the original claims." *Id*. Proxyconn may not add a proposed claim while retaining the original claim for which the proposed claim is substituted. 35 U.S.C. § 316(d)(1)(B) and 37 C.F.R. § 42.121(a)(3). Therefore, to the extent that Proxyconn's Motion to Amend requests entry of substitute claims in addition to the original claims, we deny the Motion.

[6] Proxyconn mistakenly refers to substitute claims 35–42 in its Motion to Amend, but proffers only claims 35–41 as substitutes for claims 1, 3, 6, 10, 11, 22, and 23, respectively. Mot. Amend, Appendix A.

1. Proxyconn's Burden of Persuasion Relating to Claims 35–41

Inter partes review is neither examination nor reexamination of a patent application. We do not enter proposed substitute claims as a matter of right. Rather, the patent owner must prove its entitlement to the proposed claims. Rule 42.20(c) states: "(c) *Burden of proof.* The moving party has the burden of proof to establish that it is entitled to the requested relief." 37 C.F.R. § 42.20(c). We set forth the requirements for demonstrating the prima facie patentability of substitute claims in *Idle Free Sys., Inc. v. Bergstrom, Inc.*, IPR2012-00027, Paper 26, as follows:

> A patent owner should identify specifically the feature or features added to each substitute claim, as compared to the challenged claim it replaces, and come forward with technical facts and reasoning about those feature(s), including construction of new claim terms, sufficient to persuade the Board that the proposed substitute claim is patentable over the prior art of record, and over prior art not of record but known to the patent owner. The burden is not on the petitioner to show unpatentability, but on the patent owner to show patentable distinction over the prior art of record and also prior art known to the patent owner. Some representation should be made about the specific technical disclosure of the closest prior art known to the patent owner, and not just a conclusory remark that no prior art known to the patent owner renders obvious the proposed substitute claims.
>
> A showing of patentable distinction can rely on declaration testimony of a technical expert about the significance and usefulness of the feature(s) added by the proposed substitute claim, from the perspective of one with ordinary skill in the art, and also on the level of ordinary skill, in terms of ordinary creativity and the basic skill set. A mere conclusory statement by counsel, in the motion to amend, to the effect that one or more added features are not described in any

> prior art, and would not have been suggested or rendered obvious by prior art, is on its face inadequate.

Idle Free Sys., Inc. v. Bergstrom, Inc., IPR2012-00027, slip op. at 7–8 (PTAB June 11, 2013 (Paper 26)); *see also Idle Free Sys., Inc. v. Bergstrom, Inc.*, IPR2012-00027, slip op. 33–38 (PTAB January 7, 2014 (Paper 66)).

Proxyconn has not proffered sufficient arguments or evidence to establish a prima facie case for the patentability of claims 35–41. For example, Proxyconn has not: (i) construed the newly added claim terms; (ii) addressed the manner in which the claims are patentable generally over the art; (iii) identified the closest prior art known to it; (iv) addressed the level of ordinary skill in the art at the time of the invention; or (v) discussed how such a skilled artisan would have viewed the newly recited elements in claims 35–41 in light of what was known in the art. Instead, Proxyconn attempts to distinguish claims 35–41 only from the prior art for which we instituted review of corresponding claims 1, 3, 6, 10, 11, 22, and 23. Mot. Amend 4–15. Consequently, Proxyconn has failed to establish a prima facie case for the patentability of claims 35–41. Proxyconn's motion to amend in connection with claims 35–41 is, therefore, denied on these grounds. We also find Proxyconn's motion unavailing for additional reasons raised by Microsoft as described below.

2. Patentability of Claims 35–41 in light of DRP

a. Claims 35, 36, 38, 40, and 41

Microsoft contends that DRP anticipates claims 35, 36, 38, and 40–41 and supports its contentions with detailed citations to DRP. Microsoft Corporation's Opposition to Patent Owner's Corrected Motion to Amend under 37 C.F.R. § 42.121 (Paper 48) 1–15 ("MS Amend Opp."). In its

Motion to Amend, Proxyconn does not compare claims 35, 36, 38, 40, and 41 to DRP. Mot. Amend 4–13. Proxyconn argues that Microsoft improperly injects a "new ground of review" into the trial by asserting that DRP would anticipate claims 35, 36, 38, 40, and 41 because these claims are amended versions of claims 1, 3, 10, 22, and 23 respectively, for which no challenge based on DRP has been instituted. Mot. Amend Reply 2–3.

Proxyconn's argument is unpersuasive. "Petitioners may respond to new issues arising from proposed substitute claims including evidence responsive to the amendment. 35 U.S.C. 316(a) and 326(a). This includes the submission of new expert declarations that are directed to the proposed substitute claims." Office Patent Trial Practice Guide, 77 Fed. Reg. 48,756, 48,767 (August 14, 2012).[7] Microsoft has provided evidence—i.e., specific citations to portions of DRP—that responds to new issues introduced by Proxyconn's proposed substitute claims. Microsoft is entitled to do so. Proxyconn provides no evidence to counter Microsoft's contentions that DRP anticipates claims 35, 36, 38, 40, and 41. Proxyconn carries the burden of proof with respect to the patentability of its proposed claims. 37 C.F.R. § 42.20(c). Because the only evidence of record supports Microsoft's position, we conclude that Proxyconn has failed to establish by a preponderance of evidence that claims 35, 36, 38, 40, and 41 are patentable

[7] Proxyconn's Motion to Amend was filed after publication of our decision in *Idle Free Sys., Inc. v. Bergstrom, Inc.*, IPR2012-00027, Paper 26, discussed above, which set forth the requirements for meeting the burden of proof on a motion to amend. We also reminded Proxyconn at the oral hearing of its duty to distinguish the proposed claims from all prior art of which it is aware, including DRP. Tr. 64:3–13.

over DRP. Therefore, we deny the Motion to Amend as it relates to claims 35, 36, 38, 40, and 41.

b. Claim 37

Claim 37, which is substituted for claim 6, is reproduced below with additions indicated by underlining.

> 37. A system for data access in a packet-switched network, comprising:
>
> a gateway including an operating unit, a memory and a processor connected to said packet-switched network in such a way that network packets sent between at least two other computers pass through it;
>
> a caching computer connected to said gateway through a fast local network, wherein said caching computer includes an operating unit, a first memory, a permanent storage memory and a processor; said caching computer further including a network cache memory in its permanent storage memory, means for calculating a digital digest <u>on data</u> and means for comparison between a digital digest on data in its network cache memory and a digital digest received from said packet-switched network through said gateway<u>, wherein said data includes a plurality of octet ranges in a file or files</u>.

Mot. Amend, App. A, 1–2.

We have determined that DRP anticipates claim 6. *See* Part II.C.1.d(1) above. Proxyconn's entire argument for distinguishing claim 37 from DRP is: "Substitute claim 37 requires structure to operate a data including a plurality of octet ranges in a file or files. DRP discloses content identifier based on information objects . . . . Lacking this additional element, DRP . . . fail[s] to anticipate and/or render obvious Substitute Claim 37." Mot. Amend 8–9. Proxyconn fails to cite any support for its characterization of DRP. Microsoft responds: "The data processed in DRP

includes plural files and thus includes a plurality of ranges of octets in plural files. . . . The 'data that is distributed . . . can consist of any kind of code or content.'" MS Amend Opp. 6 (citing DRP, 2:31–32, 2:44–3:2, 3:14–16, 3:28–31) (internal citation omitted). Because we agree with Microsoft, we determine that Proxyconn has failed to establish by a preponderance of evidence that claim 37 is patentable over DRP.

c. Claim 39

Claim 39, which is substituted for claim 11, is reproduced below with additions indicated by underlining and deletions indicated by double square bracketing.

> 39. A method performed by a sender/computer in a packet-switched network for increasing data access, said sender/computer including an operating unit, a first memory, a permanent storage memory and a processor and said sender/computer being operative to transmit data to a receiver/computer, the method comprising the steps of:
>
> creating a digital digest on data; [[and]]
>
> receiving a request for said data from the receiver/computer;
>
> in response to the request for data, transmitting said [[a]] digital digest of said data from said sender/computer to said receiver/computer;
>
> receiving a response signal from said receiver/computer at said sender/computer, said response signal containing a positive, partial or negative indication signal for said digital digest, and
>
> if a negative indication signal is received, transmitting said data from said sender/computer to said receiver/computer.

Mot. Amend, App., A 2–3.

Proxyconn contends, without citing any portion of DRP in support,[8] that DRP fails to describe "the step of receiving a request for data, and in response to the request for data, transmitting a digital digest for the data." Mot. Amend 10. In response, Microsoft contends that DRP describes this missing step being performed by the server responding to a GET request from the client for an index file with the current version of the index. MS Amend Opp. 10–11 (citing DRP, 5:22–32, 6:43–7:1, 7:20–31, 7:37–38, 8:8–13, 9:22–32). We determine that on the evidence before us, Proxyconn has failed to establish by a preponderance of evidence that claim 39 is patentable over DRP.

3. Alleged Broadening of Scope in Claims 36, 38, 40, and 41

During *inter partes* review, a patent owner may not amend a challenged claim in a manner that enlarges the scope of that claim. 35 U.S.C. § 316(d)(3); 37 C.F.R. § 42.121(a)(2)(ii). Proxyconn states without further discussion that "the amendments herein do not seek to enlarge the scope of the claims of the '717 Patent." Mot. Amend 1. Microsoft argues that Proxyconn's proposed claims 36, 38, 40, and 41 impermissibly enlarge the scope of challenged claims 3, 10, 22, and 23 respectively. MS Amend Opp. 4, 7. We address each of these claims in turn below.

[8] Proxyconn cites section II.F (sic, II.G) of Proxyconn's Patent Owner Response. Mot. Amend 10. However, this section of the Response cites only ¶¶ 52–61 of Dr. Konchitsky's Declaration. Resp. 36–41. None of those paragraphs cites any portion of DRP to support Dr. Konchitsky's testimony. *See* Ex. 2002, Konchitsky Decl. ¶¶ 52–61.

a. Claim 36

Claim 36, which is substituted for claim 3, is reproduced below with additions indicated by underlining and deletions indicated using strikethrough or double square bracketing.

> 36. A system for data access in a packet-switched network, comprising[[:]] a sender/computer including an operating unit, a first memory, a permanent storage memory and a processor; and a remote receiver/computer including an operating unit, a first memory, a permanent storage memory and a processor[[,]]; said sender/computer and said receiver/computer ~~communicating through said network;~~ configured to communicate with one another through said network; said sender/computer ~~further~~ includ~~es~~ing means for creating digital digests on data[[;]] stored in said permanent storage memory, and said receiver/computer ~~further~~ including a network cache memory ~~and~~, means for creating digital digests on data in said network cache memory[[;]], and ~~said receiver/computer including~~ means for ~~comparison~~ comparing between digital digests created by the sender/computer and receiver/computer; wherein said receiver/computer further includes means for storing ~~said created~~ at least one of the digital digests created by the sender/computer in its ~~first or~~ permanent storage memory; wherein the data includes at least a range of octets in a file.

See Mot. Amend, App. A, 1.

Microsoft argues that Proxyconn enlarges claim 36 in two ways. First, the claim is allegedly enlarged by changing the requirement that the sender/computer and receiver/computer are "communicating through said network" to two computers that are "configured to communicate with one another through said network." MS Amend Opp. 4. We agree that claim 36 no longer requires that the sender and receiver computers are communicating through the network, which would necessarily also require

that the computers be "configured" to so communicate. Claim 36 more broadly covers sender and receiver computers that are configured to communicate, but are not necessarily communicating. Doing so impermissibly enlarges the scope of claim 3. Therefore, we deny Proxyconn's motion to amend regarding claim 36, and need not reach Microsoft's second argument.

b. Claim 38

Claim 38, which is substituted for claim 10, is reproduced below with additions indicated by underlining and deletions indicated using strikethrough or double square bracketing.

> 38. A system for data access in a packet-switched network, comprising:
>
> a sender/computer including an operating unit, a first memory, a permanent storage memory and a processor and a remote receiver/computer including an operating unit, a first memory, a permanent storage memory and a processor, said sender/computer and said receiver/computer communicating through a network;
>
> said sender/computer further including means for creating digital digests on data, and
>
> said receiver/computer further including a network cache memory, means for storing [[a]] at least one of said digital digest received from said network in its permanent storage memory, and said receiver/computer configured to search for a digital digest received from the sender/computer, in response to receiving the digital digest, ~~means for comparison between digital digests;~~ wherein said data includes at least a range of octets in a file.

Mot. Amend, App. A, 2.

Microsoft argues that "said receiver/computer configured to search for a digital digest received from the sender/computer, in response to receiving the digital digest" covers subject matter not covered by the receiver/computer's "means for comparison between digital digests" recited in claim 10. MS Amend Opp. 6–7. Microsoft contends, "[f]or example, the receiver might check for a given digest by comparing hashes (or other identifiers) of digital digests not the digital digests themselves." *Id.* at 7. We agree. Claim 38 covers a receiver/computer that can search for a digital digest in ways other than comparing the digital digests themselves. Therefore, we deny Proxyconn's motion to amend regarding claim 38.

c. Claims 40 and 41

Claim 40, which is substituted for claim 22, is reproduced below with additions indicated by underlining and deletions indicated using strikethrough or double square bracketing.

> 40. A method for increased data access performed by a receiver/computer in a packet-switched network, said receiver/computer including an operating unit, a first memory, a permanent storage memory, a processor and a network cache memory, said method comprising the steps of:
>
> sending a request for data;
>
> receiving a message containing a digital digest for the requested data from said network;
>
> searching for each received digital digest ~~data with the same digital digest~~ in said network cache memory,
>
> if ~~data having~~ the same digital digest as the digital digest received is not uncovered, forming a negative indication signal and transmitting the negative indication signal [[it]] back through said network; and

> creating a digital digest for data received from the sender/computer and stored in said network cache memory.

Mot. Amend, Appendix A 3.

Claim 41, which is substituted for claim 23, is reproduced below with additions indicated by underlining and deletions indicated using strikethrough.

> 41. The method as claimed in ~~claim 22,~~ claim 40, wherein searching in said network cache memory includes ~~further comprising~~ searching in predetermined locations in said permanent storage memory for data with a digital digest ~~substantially~~ identical to the searched one of the digital digests received from said network.

Id.

Microsoft argues that the third and fourth changes to claim 22 reflected in proposed claim 40 impermissibly enlarge the scope of claim 22. Microsoft contends that "[c]laim 40 newly covers methods that look only for matching digests but not for data having those digests. Claim 22 does not cover such a method." MS Amend Opp. 11. Proxyconn responds that "searching specifically for 'each received digital digest,' . . . is narrower than merely searching generally for data by the digital digest." Mot. Amend Reply 5. Proxyconn's argument mischaracterizes claim 22 as requiring "searching generally for data by the digital digest." Claim 22 recites: "searching for data with the same digital digest" not "by the same digest."

The question presented by the parties' arguments is whether it is possible to search for each received digital digest without searching for data having the same digital digest. Microsoft does not identify an example of how one might search for data with the same digest without using the digest. However, Proxyconn does not provide evidence that it is not possible to

search for each received digest without searching for data having the same digest. Without such evidence, we can only evaluate the scope of the claim based on the plain meaning of the terms. On that basis, we conclude that it would be possible to search for each received digital digest without searching for data having the same digital digest. Therefore, it is possible to practice the method recited in claim 40 without practicing the method recited in claim 22. For this reason, we conclude that claim 40 is impermissibly broader than claim 22. The same flaw exists in claim 41, which depends from claim 40. Therefore, we deny Proxyconn's Motion to Amend as it relates to claims 40 and 41.

III. CONCLUSION

Microsoft has established by a preponderance of evidence that claims 1, 3, 6, 7, 9, 10, 11, 12, 14, 22, and 23 are unpatentable as anticipated and claims 1, 3, and 10 are unpatentable as being directed to obvious subject matter. Microsoft has not established by a preponderance of evidence that claim 24 is unpatentable.

IV. ORDER

It is ORDERED that:

Claims 1, 3, 6, 7, 9–12, 14, 22, and 23 are CANCELED; and

Proxyconn's Motion to Amend Claims is DENIED.

PATENT OWNER:

Matthew L. Cutler
Bryan K. Wheelock
Douglas A. Robinson
HARNESS, DICKEY & PIERCE, PLC
mcutler@hdp.com
bwheelock@hdp.com
drobinson@hdp.com

PETITIONER:

John D. Vandenberg
Stephen J. Joncus
KLARQUIST SPARKMAN LLP
john.vandenberg@klarquist.com
stephen.joncus@klarquist.com