

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

DROPBOX, INC. AND MICROSOFT CORPORATION,
Petitioners,

v.

DATANET, LLC,
Patent Owner.

IPR2024-00079
U.S. Patent No.: 9,218,348
Issued: December 22, 2015
Application No.: 13/925,768
Filed: June 24, 2013

Title: Automatic Real-Time File Management Method And Apparatus

PETITION FOR
INTER PARTES REVIEW OF U.S. PATENT NO. 9,218,348

TABLE OF CONTENTS

	Page(s)
LIST OF EXHIBITS	vii
I. MANDATORY NOTICES UNDER 37 C.F.R. § 42.8.....	ix
1. Real Party-In-Interest.....	ix
2. Related Matters	ix
3. Lead And Back-Up Counsel, And Service Information.....	x
I. INTRODUCTION	1
II. GROUNDS FOR STANDING PER 37 C.F.R. § 42.104(a)	1
III. IDENTIFICATION OF CHALLENGE	2
A. Statement Of The Precise Relief Requested/Statutory Grounds.....	2
B. No Examiner Addressed These Unpatentability Grounds	2
C. The Board Should Institute Despite Parallel Litigation	3
IV. STATE OF THE ART	3
V. THE '348 PATENT	4
A. The '348 Specification	4
B. The Prosecution History	5
C. Claims Listing	6
VI. LEVEL OF SKILL IN THE ART	6
VII. CLAIM CONSTRUCTION	7
VIII. GROUND 1: CLAIMS 1-6, 8-20, 23-31 OBVIOUS OVER KENLEY AND LONG.....	7
A. Kenley (EX1014)	7

B.	Long (EX1004).....	8
C.	Combining Kenley And Long	13
D.	Claim Mapping.....	19
1.	Claim 1	19
2.	Claim 2.....	40
3.	Claim 3	41
4.	Claim 4.....	43
5.	Claim 5.....	44
6.	Claim 6.....	46
7.	Claim 8.....	47
8.	Claims 9-11	48
9.	Claims 12-15.....	48
a)	Element [12.6]	49
b)	Element [13.6]	50
c)	Element [14.6]	51
d)	Element [15.6]	52
10.	Claim 16.....	53
11.	Claim 17.....	53
12.	Claim 18.....	53
13.	Claim 19.....	54
14.	Claim 20.....	55
15.	Claim 23.....	55
16.	Claim 24.....	55

17.	Claim 25	55
18.	Claim 26	57
19.	Claim 27	57
20.	Claim 28	58
21.	Claim 29	58
22.	Claim 30	59
23.	Claim 31	59
IX.	GROUND 2: CLAIMS 1-6, 8-20, AND 23-31 ARE OBVIOUS OVER CABRERA, SCHNEIDER, AND WHITING.....	59
A.	Cabrera (EX1005)	59
B.	Schneider (EX1020)	62
C.	Whiting (EX1006).....	62
D.	Combining Cabrera, Schneider, And Whiting	63
E.	Claim Mapping	66
1.	Claim 1	66
a)	Element [1.1]	66
b)	Element [1.2]	66
c)	Element [1.3]	68
d)	Element [1.4]	69
e)	Element [1.5]	69
f)	Element [1.6]	71
g)	Element [1.7]	72
h)	Element [1.8]	73

	i) Elements [1.9]-[1.10].....	74
2.	Claim 2.....	74
3.	Claims 3-5	74
4.	Claim 6.....	75
5.	Claim 8.....	76
	a) Elements [8.7]-[8.8].....	76
	b) Elements [8.9]-[8.10].....	76
6.	Claims 9-11	77
7.	Claims 12-15	77
	a) Element [12.6]	77
	b) Element [13.6]	78
	c) Element [14.6]	78
	d) Element [15.6]	78
8.	Claim 16.....	78
9.	Claim 17.....	79
10.	Claim 18.....	79
11.	Claim 19.....	79
12.	Claim 20.....	80
13.	Claim 23	80
14.	Claim 24	80
15.	Claim 25.....	81
16.	Claim 26.....	81
17.	Claim 27	82

18.	Claim 28	82
19.	Claim 29	83
20.	Claim 30	83
21.	Claim 31	83
X.	NO OBJECTIVE INDICIA OF NON-OBVIOUSNESS	83
XI.	CONCLUSION.....	83
	CERTIFICATE OF COMPLIANCE.....	85
	CERTIFICATE OF SERVICE	86

TABLE OF AUTHORITIES

Page(s)

Statutes

35 U.S.C. § 102 7, 62

35 U.S.C. § 103 2

Other Authorities

USPTO Memorandum, Interim Procedure for Discretionary Denials in AIA Post-Grant Proceedings with Parallel District Court Litigation, (June 21, 2022).....3

Rules

37 C.F.R. § 42.24 85

LIST OF EXHIBITS

No.	Description
1001	U.S. Patent No. 9,218,348 (“ 348 patent ”)
1002	File History of U.S. Patent No. 9,218,348 (“ 348 FH ”)
1003	Declaration of David Maier, dated October 31, 2023 (“ Maier Decl. ”)
1004	U.S. Patent No. 5,325,519 (“ Long ”)
1005	U.S. Patent No. 5,953,729 (“ Cabrera ”)
1006	U.S. Patent No. 5,778,395 (“ Whiting ”)
1007	Marc Farley, <i>Building Storage Networks</i> (2000) (excerpts) (“ Farley ”)
1008	Helen Custer, <i>Inside the Windows NT File System</i> (1994) (excerpts) (“ Custer ”)
1009	Newton’s Telecom Dictionary (2000) (excerpts)
1010	U.S. Patent No. 5,860,124 (“ Matthews ”)
1011	Ann Chervenak, et al., <i>Protecting File Systems: A Survey of Backup Techniques</i> (“ Chervenak ”)
1012	U.S. Patent No. 5,485,606 (“ Midgley ”)
1013	U.S. Patent No. 5,745,313 (“ Sliger ”)
1014	U.S. Patent No. 5,276,867 (“ Kenley ”)
1015	Abraham Silberschatz, et al., <i>Operating System Concepts</i> (3d ed 1991) (“ Silberschatz ”)
1016	U.S. Patent No. 6,016,553 (“ Schneider-US ”)
1017	Joost S.M. Verhofstad, <i>Recovery Techniques for Database Systems</i> , Computing Surveys (June 1978) (“ Verhofstad ”)
1018	U.S. Patent No. 5,771,354 (“ Crawford ”)
1019	Microsoft Press Computer Dictionary (1994) (excerpts)

No.	Description
1020	PCT Publication No. WO 99/12101 (“ Schneider ”)
1021	U.S. Patent No. 5,684,991 (“ Malcolm ”)
1022	Russell Sandberg, et al., <i>Design and Implementation of the Sun Network Filesystem</i> , in <i>Innovations in Internetworking</i> (Craig Partridge ed., Artech House 1988) (excerpts) (“ Sandberg ”)
1023	List of Challenged Claims

I. MANDATORY NOTICES UNDER 37 C.F.R. § 42.8

1. Real Party-In-Interest

The real-parties-in-interest are Dropbox, Inc. and Microsoft Corporation.

2. Related Matters

The '348 patent (EX1001) has been asserted in the following litigations:

- *Datanet LLC v. Dropbox, Inc.*, No. 6:22-cv-01142 (W.D. Tex.), filed October 31, 2022; and
- *Datanet LLC v. Microsoft Corporation*, No. 2:22-cv-01545 (W.D. Wash.), filed October 31, 2022.

Two patents that allege priority to the same original non-provisional and provisional applications to which the '348 patent alleges priority are being challenged in the following IPRs:

- IPR2024-00078 challenges U.S. Pat. No. 8,473,478 (“’478 patent”); and
- IPR2024-00080 challenges U.S. Pat. No. 10,585,850 (“’850 patent”).

A patent that alleges priority to the same original non-provisional and provisional applications to which the '348 patent alleges priority is being challenged in *ex parte* Reexamination No. 90/015,216.

3. Lead And Back-Up Counsel, And Service Information

Lead Counsel	Back-up Counsel
Andrew M. Mason, Reg. No. 64,034 andrew.mason@klarquist.com	Cameron D. Clawson, Reg. No. 73,509 cameron.clawson@klarquist.com Sarah E. Jelsema, Reg. No. No. 70,804 sarah.jelsema@klarquist.com Frank Morton-Park, Reg. No. 80,750 frank.morton-park@klarquist.com
KLARQUIST SPARKMAN, LLP 121 SW Salmon Street, Suite 1600 Portland, Oregon, 97204 503-595-5300 (phone) 503-595-5301 (fax)	

Petitioners consent to service via email at the above email addresses.

Pursuant to 37 C.F.R. § 42.10(b), concurrently filed with this Petition is a Power of Attorney executed by Petitioners and appointing the above counsel.

I. INTRODUCTION

Dropbox, Inc. and Microsoft Corporation (“Petitioners”) respectfully request *inter partes* review (“IPR”) of claims 1-6, 8-20, and 23-31 of U.S. Patent No. 9,218,348 (EX1001, “’348 patent”), allegedly assigned to Datanet, LLC (“Patent Owner”). For the reasons set forth below, these claims should be found unpatentable and cancelled.

The challenged claims of the ’348 patent relate to a method for archiving files in which an operation performed on a file causes a backup copy of the file to be created and stored in temporary storage, intermediate storage, and a permanent archive storage. *E.g.*, Claim 1. This concept was already taught, however, almost a decade earlier by Long (EX1004) and became conventional years before the alleged invention, as evidenced by Cabrera (EX1005). Each of Long and Cabrera, in combination with other references, renders each challenged claim obvious.

II. GROUND FOR STANDING PER 37 C.F.R. § 42.104(a)

Petitioners certify that the ’348 patent is available for IPR and that Petitioners are not barred or estopped from requesting an IPR challenging the patent claims on the grounds identified in this petition.

III. IDENTIFICATION OF CHALLENGE

A. Statement Of The Precise Relief Requested/Statutory Grounds

Petitioners request *inter partes* review of claims 1-6, 8-20, and 23-31 (the “Challenged Claims”) of the ’348 patent, on the following statutory grounds:

	Reference(s)	Basis	Claims
Ground 1	Long (EX1004) and Kenley (EX1014)	35 U.S.C. § 103	1-6, 8-20, 23-31
Ground 2	Cabrera (EX1005), Schneider (EX1020), and Whiting (EX1006)	35 U.S.C. § 103	1-6, 8-20, 23-31

For each ground, in Sections VIII-IX below, the petition presents evidence of unpatentability and establishes a reasonable likelihood that Petitioners will prevail in establishing that each Challenged Claim is unpatentable.

An electronic payment of \$50,500.00 for the requisite fees is being charged to deposit account no. 02-4550. Any fee adjustments may be applied to that account.

B. No Examiner Addressed These Unpatentability Grounds

Prior art references Long, Kenley, Cabrera, Schneider, and Whiting were not discussed on the record during prosecution and nothing suggests that their teachings relied on herein—much less the combination of those teachings as informed by expert testimony—were considered by the Examiner. Accordingly, this petition should not be denied under Section 325(d).

C. The Board Should Institute Despite Parallel Litigation

Each Petitioner stipulates that, if the Board institutes IPR, it will not pursue in its parallel district court proceeding the grounds raised, or which reasonably could have been raised, in this petition. When a petitioner presents such a stipulation, “the PTAB will not discretionarily deny institution in view of parallel district court litigation.” USPTO Memorandum, Interim Procedure for Discretionary Denials in AIA Post-Grant Proceedings with Parallel District Court Litigation at 3, 7 (June 21, 2022).

IV. STATE OF THE ART

File-based backup systems were well-known. EX1003, ¶¶35-37. Further, by the 1990s Network File Systems (e.g., Sun’s NFS) had erased any difference between how users accessed local and remote files. EX1022, 370-390 (showing first release); EX1018, 3:64-66 (“Drives on a remote system defined using NFS ... are available ... on the Internet.”). For backup applications, there was no significant technical distinction between local and remote storage of files. EX1005, 1:38-42, 3:3-7, 6:8-28; EX1003, ¶¶38-39.

V. THE '348 PATENT

The '348 patent, issued December 22, 2015, from an application filed June 24, 2013, and alleges priority to an application filed September 21, 2001, and a provisional application filed September 21, 2000. *Id.*

A. The '348 Specification

The '348 patent describes a method “that captures files just before and/or just after the files are changed.” *Id.*, 3:9-11. It describes a computing device with input buffer, output buffer, and database; and a storage device that “may be either internal or external to computing device 5.” *Id.*, 5:1-7; FIG. 1.

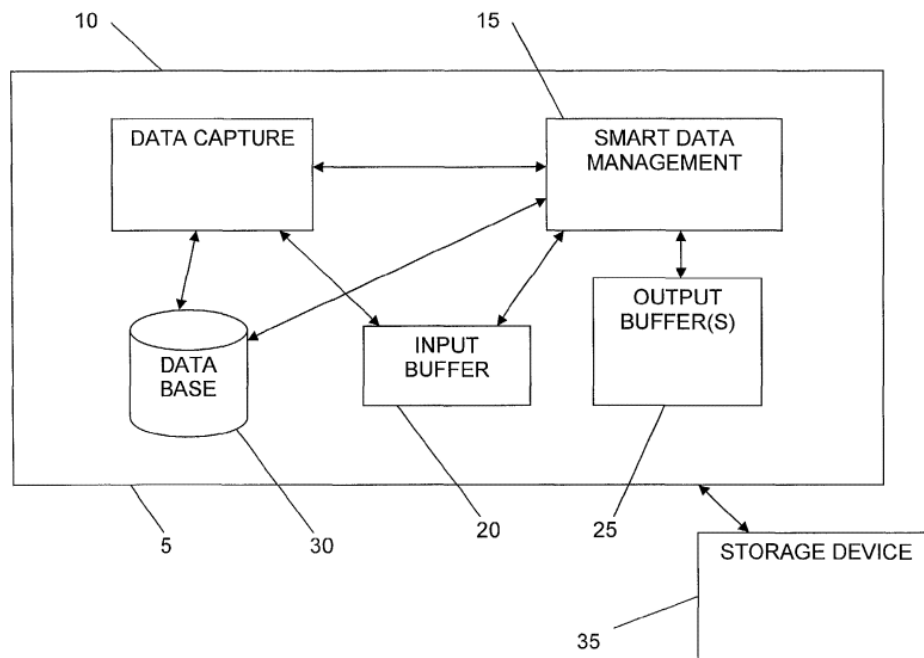


FIGURE 1

EX1001, FIG. 1.

The '348 patent describes this system in largely functional terms as detecting an instruction to operate on an operating file, “capturing” the operating file, and migrating the file “from the input buffer 20 through the output buffers 25 to storage device 35.” EX1001, 5:9-41. It describes events, such as messages from timers, triggering actions, such as examining input buffer “for the presence of archive files” or moving the archive file to a storage device. *Id.*, 5:44-51; *id.*, 5:53-6:7.

B. The Prosecution History

The first two Office actions rejected all pending claims for various reasons, including double-patenting and unpatentability over the prior art. In response to each rejection, applicants canceled all pending claims and added an entirely new set of claims, ultimately resulting in pending claims 63-82. EX1002, 271-284/369, 258-263/369; *id.*, 242-251/369, 217-223/369.

A third Office action rejected claims 63-82 for nonstatutory double patenting and as obvious over various prior art combinations. *Id.*, 147-172/369. Applicants then amended claim 63, cancelled claims 64, 80, and 82, and added new claims 83-96. *Id.*, 132-142/369.

After an Interview, the Examiner allowed the claims, which issued as claims 1-31 of the '348 patent. *Id.*, 23-26/369.

C. Claims Listing

Challenged claim 1 is representative and is listed here, with element numbering added in the lefthand column:

[1.1]	A method for archiving files, comprising steps of (a) to (i) following:
[1.2]	(a) the step of detecting an instruction by a resident program in a computing device for performing an operation on an operating file;
[1.3]	(b) the step of creating an archive file from the operating file and storing the archive file in a temporary storage location temporally proximate to the operation being performed on the operating file and responsive to detecting the instruction;
[1.4]	(c) the step of identifying presence of the archive file in the temporary storage location responsive to the occurrence of a first event;
[1.5]	(d) the step of transmitting the archive file to a second storage location responsive to a second event, the second storage location being an intermediate or a permanent storage location;
[1.6]	(e) after storing the archive file in the temporary storage location, updating a database to indicate that the archive file is located in the temporary storage location;
[1.7]	(f) determining a final destination for the archive file;
[1.8]	(g) moving the archive file from the temporary storage location to an intermediate storage location;
[1.9]	(h) updating the database to indicate the archive file is located in the intermediate storage location; and
[1.10]	(i) after moving the archive file to the second storage location, updating the database to indicate the archive file is located in the second storage location.

A full claims listing is submitted as Exhibit 1023.

VI. LEVEL OF SKILL IN THE ART

The person of ordinary skill in the art in September 2000 (“POSITA”) would have held a bachelor’s degree in computer science, electrical engineering, computer

engineering, or a closely related field, and one or more years of experience in the design and development of file management, restoration, and archival systems. More education, e.g., a Master's or Ph.D., could compensate for less work experience and vice versa. There would be no meaningful distinction if the critical date were 1999 or 2001. EX1003, ¶¶29-32.

VII. CLAIM CONSTRUCTION

Unless otherwise expressly discussed below, Petitioners apply the plain and ordinary meaning of all claim terms, without waiving any rights to raise additional issues of claim construction, indefiniteness, or invalidity in any litigation.

VIII. GROUND 1: CLAIMS 1-6, 8-20, 23-31 OBVIOUS OVER KENLEY AND LONG

A. Kenley (EX1014)

U.S. Patent No. 5,276,867 (EX1014, “Kenley”) issued January 4, 1994, and is prior art under at least pre-AIA¹ Sections 102(a-b). Kenley is analogous art to

¹ For purposes of this Petition, Petitioners assume that the pre-AIA version of the Patent Statute applies to the challenged claims, without conceding that those claims are entitled to a pre-AIA priority date.

the '348 patent because they are both within the same field of endeavor of data management and preservation. EX1001, 1:10-14; EX1014, 1:13-15; EX1003, ¶40.

Kenley describes a “hierarchical storage configuration” that stores data organized as files in three storage elements of varying capacity and access speed: a primary storage; a secondary storage; and a “backing store.” EX1014, 2:44-55; *id.*, 4:54-66. “A file storage element in the secondary storage element maintains data in the form of files” (*id.*, 3:1-4) and “the backing store stores data organized as files” (*id.*, 3:29-30). Kenley teaches migrating data from secondary storage to the backing store, either at selected times or upon receiving a storage full message. *Id.*, 2:56-68. It further teaches a relational database with a “system catalog” for maintaining and providing transparent access to file attribute and location data. *Id.*, 6:36-40, 3:17-28; EX1003, ¶41.

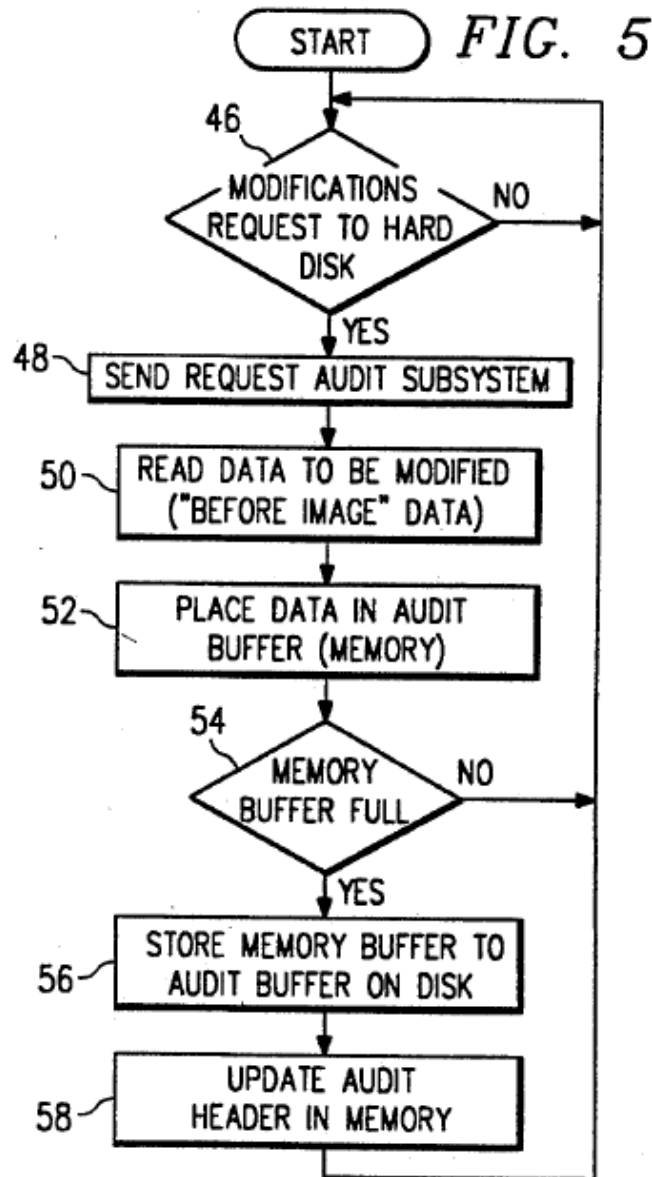
B. Long (EX1004)

U.S. Patent No. 5,325,519 (EX1004, “Long”) issued June 28, 1994, and is prior art under at least pre-AIA Sections 102(a-b). Long is analogous art to the '348 patent because they are both within the same field of endeavor of data management and preservation. EX1001, 1:10-14; EX1004, 1:5-7, 1:56-59; EX1003, ¶42.

Long teaches a general-purpose backup technique that creates a backup copy of data prior to modification of that data, and then archives that data to separate storage. Long teaches that its technique is broadly “[a]pplicable to any mass storage

media wherein old data may be overwritten with new data.” EX1004, 9:54-56. EX1003, ¶¶43-44.

To preserve historic versions of data, Long teaches creating copies prior to any changes or modifications to data, based on a trigger such as an application issuing a “modification request ... to modify the contents of the hard disk.” *Id.*, 7:28-29, 7:33-34. Then, “before the data on the disk is modified,” “the request is sent to the audit subsystem,” which triggers a backup process. *Id.*, 7:33-37, FIG. 5. EX1003, ¶¶45-46.



EX1004, FIG. 5.

The “modification request” leads to a backup copy (“before image” data) being created and stored in a “memory buffer.” *Id.*, 7:43-53. The backup is then moved from the memory buffer to an “audit buffer” on the local disk. *E.g., id.*, 7:38-43. The process then updates an audit header with “information that is used to manage the audit buffer 30, archive media ... and the general environment of the

audit subsystem.” *Id.*, 4:44-47. Thus, Long creates a backup that “contains the data that existed on one sector of the disk prior to the write occurring.” *Id.*, 6:46-48. EX1003, ¶46.

Long further teaches copying the data from the local disk audit buffer to a separate archive media on “external storage.” *E.g., id.*, 8:36-9:9 (describing the “[a]rchiving of the audit buffer”). The archive media is “external storage” on a “removable media device,” such as a floppy disk. *Id.*, 7:12-15, 8:32-33, 2:48-49. Long describes an archive method that “allows the archiving to take place in the background so that the user may continue processing during an archive operation.” *Id.*, 8:42-45. This archive process, which works together with the audit buffer backup process, is reflected in Long Fig. 6:

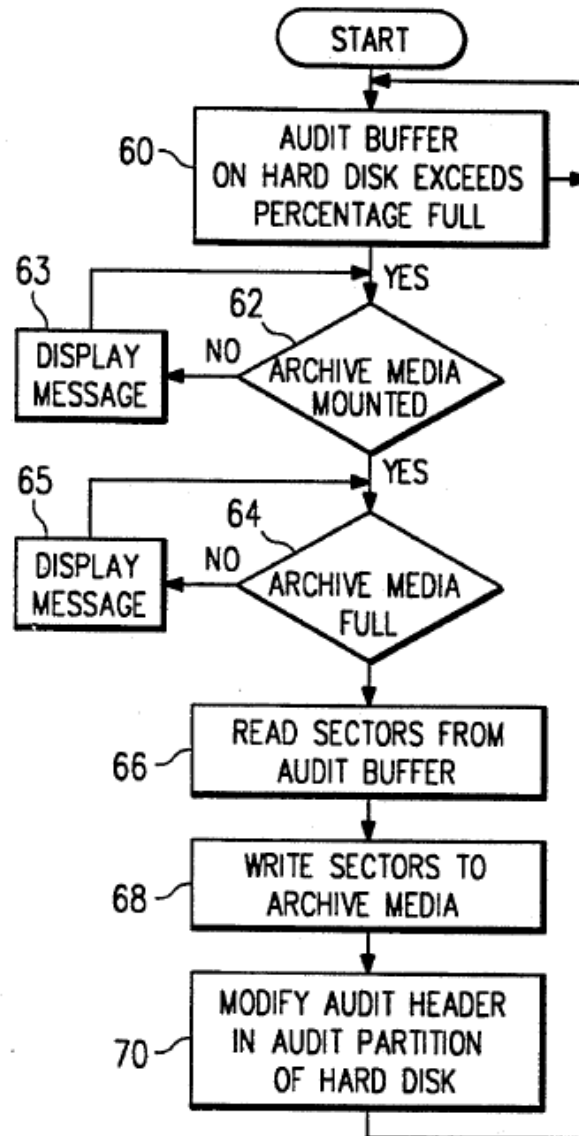


FIG. 6

EX1004, FIG. 6. EX1003, ¶47.

Long also teaches modifying the audit header to reflect transfer of the data to the archive media. *Id.*, 9:7-9. This allows “rollback ... to a previous known state.” *Id.*, 9:10-13. EX1003, ¶48.

C. Combining Kenley And Long

A POSITA would have been motivated, with a reasonable expectation of success to implement Kenley’s file-based backup system with the modification-driven archival approach taught by Long, to improve reliability and provide other benefits to the system. The “incremental backup” technique of Kenley provides only “[a] copy of ... specified files of data that has been modified since a specified date, usually the date of the last full or incremental backup.” *E.g.*, EX1014, 7:19-30, 5:3-6. Long, in turn, notes the data risks posed by power outages and operator error (EX1004, 1:46-55), and provides “significant improvements in hard disk error recovery” (*id.*, 1:62-65). These improvements include increased reliability by creating backup data after each modification, thus avoiding the potential loss of data created in between the timed backups of Kenley and other similar systems. *Id.*, 4:23-27. This combination both serves the goal of “reduc[ing] the time associated with conventional data migration schemes” taught by Kenley, and provides an “efficient, reliable and cost[-]effective architecture for disk fault tolerance” as taught by Long. EX1014, 2:26-30; EX1004, 2:13-14. As explained in the following paragraphs, for at least the following reasons, a POSITA would have made this combination. EX1003, ¶¶49-51.

First, Long teaches a “Fault Tolerant Computer With Archival Rollback Capabilities” that “includes significant improvements in hard disk error recovery”

and “provides a[n] efficient, reliable and cost[-]effective architecture for disk fault tolerance.” EX1004, 1:2-3, 1:62-64, 2:13-14. A POSITA would have thus been motivated to improve Kenley’s file-based “hierarchical digital data storage system” (EX1014, 2:33-36, 3:1-47) by adding Long’s event-driven backup process that efficiently saved backups prior to any modifications of a file stored on hard disk. EX1003, ¶52.

Second, a POSITA knew that Long’s teachings were applicable to files. Long expressly states that its “invention is applicable to any mass storage media wherein old data may be overwritten with new data” and broadly discusses “data on a hard disk” and the copying of such data prior to modifying “the contents of the hard disk.” EX1004, 10:54-56, 1:46-48, 4:23-25, 7:28-29. Thus, a POSITA reading Long would immediately contemplate use of its techniques within file-based systems, for preserving data stored in files within that system. *See also* EX1011, 19 (“Backup software can operate either on files or on physical disk blocks.”); EX1005, 1:32-34 (the “prior state of the art” included archive systems that “maintain[ed] a complete history of the changes made to ... a particular file”). The POSITA knew that data on a hard disk was most commonly organized as files to facilitate data access using a file system. *E.g.*, EX1013, 5:18-51; EX1012, 1:27-60. Indeed, the background section of one background reference flatly states what was well-known: “[i]nformation processed by a computer is organized into distinct units called files.”

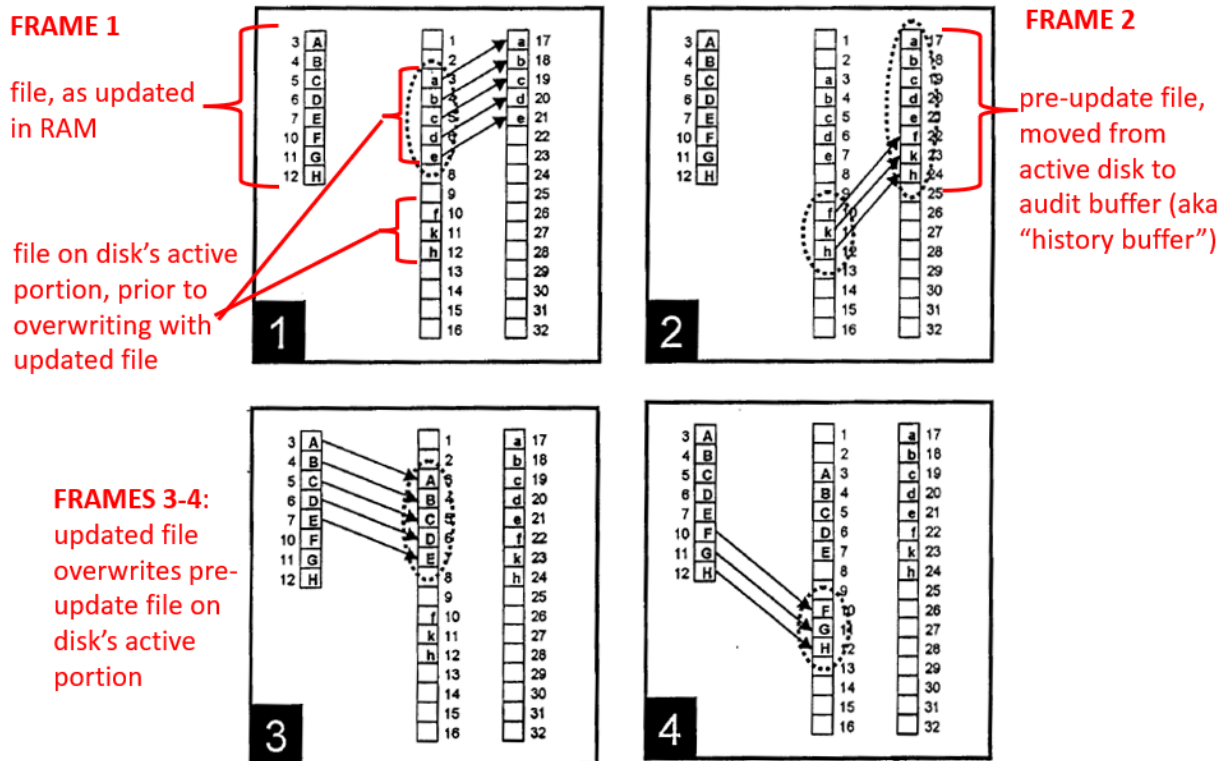
EX1012, 1:26-27. Moreover, it was well-known that applications in file-based systems would perform write and other operations that modified the file. *E.g.*, EX1010, 1:40-43. This is reinforced by Kenley, which describes “stor[ing] data organized as files” and using file attributes to track the backup location of files enables the data to be more easily and transparently recovered from archive storage media for use by an application. EX1014, 3:29-32, 7:19-62. EX1003, ¶53.

Third, the two references are complementary, as each teaches backup systems with at least three tiers of storage—primary storage (e.g., RAM or semiconductor memory buffer), secondary storage (e.g., hard disk), and archive or backing storage (e.g., external storage). EX1014, 2:44-55; EX1004, 7:43-53 (copying data to memory buffer and then to audit buffer on disk), 8:29-9:9 (moving data from audit buffer on disk to archive media). Kenley also teaches “‘transparent’ access to digital data” stored in archive storage media, with “access that entails no significant delay in ongoing processes” and “a storage system interface that communicates with applications programs in a manner functionally identical to that of the existing system interface.” EX1014, 2:33-43. This “transparent” archiving aligns with Long’s teachings on a system that “allows the archiving to take place in the background so that the user may continue processing during an archive operation.” EX1004, 8:42-49; EX1003, ¶54.

Fourth, Kenley’s teaching of a database-held system catalog that “stor[es] and provid[es] transparent access to catalogued file attribute data” (EX1014, 3:17-24) dovetails with Long’s teachings on an audit header for similarly maintaining data on the current location of backup copies of data (*e.g.*, EX1004, 7:47-53). Kenley’s system catalog “enhances transparent access to file attribute data, independent of whether the file is stored in the secondary storage element or has been previously migrated from the secondary storage element” to the backing store. EX1014, 3:25-28. Thus, a POSITA would naturally augment the catalog database of Kenley to store relevant information from the audit header taught by Long, thus “enhanc[ing] transparent access.” *Id.*; *see also* EX1004, 7:47-53, 9:7-9 (updating audit header to reflect changes after migrating data); EX1014, 6:35-40 (system catalog is part of a database), 8:26-29 (updating the database with catalog of backed up files), 10:43-52 (using the database backup catalogs to locate backup files); EX1003, ¶55.

Fifth, Long teaches a straightforward modification-driven archival approach that applied to known multi-tier backup-archival systems. EX1004, 11:54-56. A POSITA would have easily applied that approach in a file-based system. As explained in mapping the claims below, this method uses the same event-driven approach as the Challenged Claims. *Infra* Section VIII.D. Thus, a POSITA had a reasonable expectation of success in implementing the methods of the Challenged Claims. EX1003, ¶56.

The reasonable expectation of success, and overall obviousness, of implementing Long's technique for a file-based system such as Kenley's is confirmed by another 1998-filed prior-art patent that expressly describes using Long's audit buffer technique to save files before alteration. EX1016, 46:36-47:8. That background reference reflects the knowledge of a POSITA and shows in FIG. 37 "what normally happens when data is written." *Id.* It then contrasts that with the Long technique (aka "Move" method) in FIG. 38, which saves a file just before an overwrite. *Id.*; see also *id.*, 46:25-26; *id.*, 6:33-36.



Id., FIG. 38 (annotated based on description at 46:36-47:9). This confirms a POSITA would have naturally and easily implemented Long’s method for file-based systems. EX1003, ¶57.

So implemented, the resulting Kenley-Long system would, based on computer-executable code within the system, perform at least these operations:

- Generate a modification request, as taught by Long, any time the operating system or a user application “need[ed] to modify the contents of” a file stored on the hard disk. For example, a text-editing program issuing a write operation for a file would be received by the computer system and then “sent to the audit subsystem.” EX1004, 7:27-53.
- Carry out the audit and archive technique taught by Long, on the file being modified, including the copying of files first to the memory buffer, second to the audit buffer portion of the hard drive, and then to external storage, as taught by Long, thus creating “archive copies” of files within a file-based system as taught by Kenley. EX1014, Abstract.
- Maintain relevant audit header information, as taught by Long, in a backup catalog in a relational database, as taught by Kenley. EX1003, ¶¶55, 58.

D. Claim Mapping

1. Claim 1

[1.1]² A method for archiving files, comprising steps of (a) to (i) following:³

To the extent the preamble is limiting, Kenley-Long satisfies it. Long teaches a backup and archiving process “with archival rollback capabilities,” as reflected in its flow charts illustrating the archiving of data. EX1004, FIGS. 4-6; *id.*, 2:28-33, 12:19-37 (Claim 31). When used to implement a file-based system such as Kenley’s, the Long process archives files. EX1003, ¶59; *supra* Section VIII.C.

[1.2] (a) the step of detecting an instruction by a resident program in a computing device for performing an operation on an operating file;

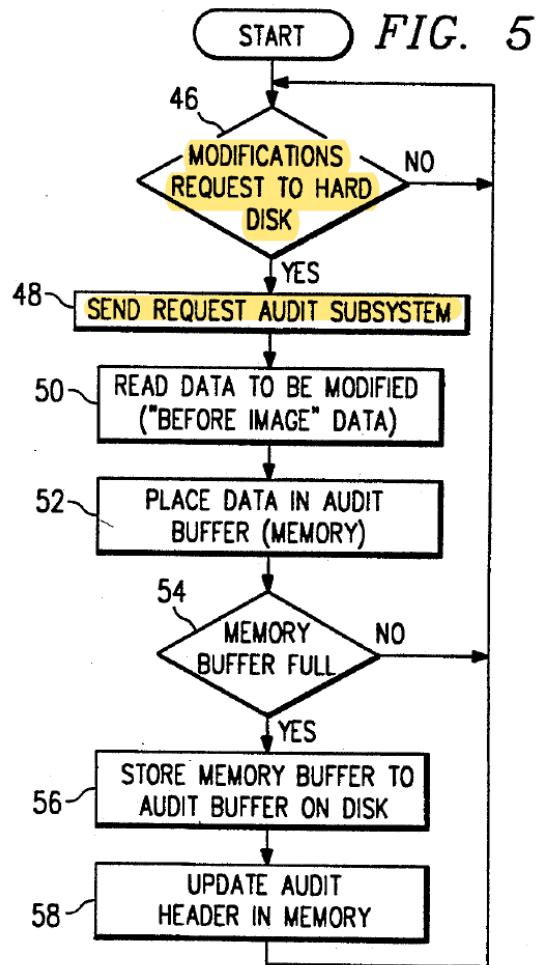
Long teaches “detecting access requests to alter data on respective sets of one or more locations of the storage device.” EX1004, Abstract. Long specifies that “[p]rocessing circuitry detects accesses to the [storage] device which would alter data thereon.” EX1004, 1:67-68. Long further discloses that “[a] ‘modification request’ is a request from an application to modify the contents of the hard disk.” EX1004, 7:28-29, FIG. 5. A modification request is generated “[w]hen the computer

² Reference numbers are added throughout in the format [claim#.limitation#].

³ To the extent antecedent basis requires an order of certain steps, that is addressed in mapping the claim elements.

system 10 needs to write to a hard disk.” *Id.*, 7:32-37. Thus, Long discloses detecting an instruction by a resident program in a computing device for performing an operation on data. EX1003, ¶60.

When used to implement a file-based system such as Kenley’s, the Long audit subsystem process would detect instructions from user applications and the operating system for “performing an operation on an operating file.” *Supra* Section VIII.C. It was well known for applications and operating systems to request modifications to a file, e.g., write, creation, or deletion operations. EX1004, 1:48-53 (describing write operation scenario); *id.*, 4:23-25; *see also* EX1006, 1:11-13 (“data and program files” are “often together referred to as ‘data’”); EX1008, 62 (data of a “normal” file); EX1005, 6:65-7:2 (data is traditionally stored in a file format); EX1010, 1:13-17 & 1:40-43; EX1012, 1:26-35. In the Kenley-Long system, any of these file modification requests would be detected by the audit subsystem. The operating system was a resident program by definition of the ’348 patent, and user applications were resident programs at least because they have control over file operations such as read, write, open, and save. *See* EX1001, 4:49-52 (defining “Resident Program”). Long’s audit subsystem is also a resident program as it is “part of the system BIOS” or a “memory-resident program.” EX1004, 6:60-63. EX1003, ¶61.

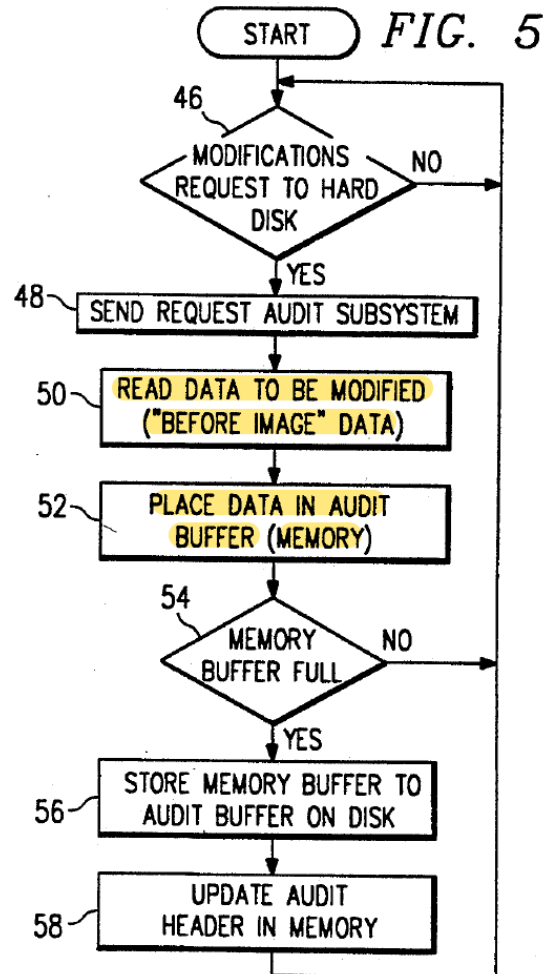


EX1004, FIG. 5 (highlighting added).

Kenley-Long thus renders obvious “the step of detecting an instruction by a resident program in a computing device for performing an operation on an operating file.” EX1003, ¶62.

[1.3] (b) the step of creating an archive file from the operating file and storing the archive file in a temporary storage location temporally proximate to the operation being performed on the operating file and responsive to detecting the instruction;

Kenley discloses storing “archive copies” of “data files.” EX1014, Abstract. Long teaches creating a copy of the data to be modified and storing it in a temporary storage location, responsive to detecting the modification request. For example, the actions “in response to a modification request” include (1) reading the data to be modified, i.e., the “before image” data, from the disk and (2) storing that data in “the memory buffer.” EX1004, 7:29-40, 7:43-46, 7:54-55, 7:59-64. These actions occur “temporally proximate” to the modification of the operating file at least because they (1) are responsive to the modification request and (2) the “requested write” or other operation (i.e., the modification) is made to the hard disk just after the actions are completed. *See* EX1004, 8:14-17, 7:33-40; EX1001, 5:12-17 (defining “temporally proximate” as “just before and/or just after”). Long shows exemplary operations in FIG. 5:



EX1004, FIG. 5 (highlighting added). EX1003, ¶63.

When implemented for a file-based system such as Kenley's, these actions taught by Long result in a copy of the operating file being created and stored as a file in the memory buffer (temporary storage) temporally proximate to the operating file being modified. Kenley teaches storing data in the form of files, and thus a POSITA using Long's method to archive files would have initially stored the copy of the operating file (i.e., the archive file) in the memory buffer as taught by Long. EX1014, 3:1-2 ("the secondary storage maintains data in the form of files"), 3:29-

30 (“the backing store stores data organized as files”); *supra* Section VIII.C. This copy of the file is an archive file at least because it is a file that captures all of the data of an operating file. EX1001, 4:17-21 (defining “archive file” as including “a file containing all of the data of an operating file in a native or altered format ...”); *see also* EX1014, Abstract (describing storing “archive copies” of “data files”). EX1003, ¶64.

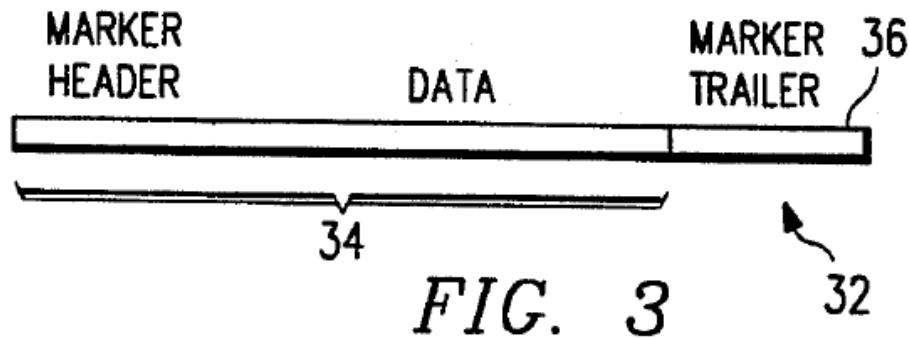
Thus, Kenley-Long renders obvious “creating an archive file from the operating file and storing the archive file in a temporary storage location temporally proximate to the operation being performed on the operating file and responsive to detecting the instruction.” EX1003, ¶65.

[1.4] (c) the step of identifying presence of the archive file in the temporary storage location responsive to the occurrence of a first event;

Long teaches identifying the presence of the archived data in the memory buffer, responsive to a “first event,” in at least two ways. It teaches events that include (i) the memory buffer filling up and (ii) the “processing unit 14 requir[ing] memory used in the memory buffer for another purpose.” EX1004, 7:47-53. A POSITA knew that most obviously this information would arise in the form of messages from the operating system or some other resident program such as the BIOS. Either event triggers the audit subsystem identifying the data’s presence in the memory buffer as reflected by the fact that, responsive to either of these two

events, this data is moved to the audit buffer. EX1004, 8:20-23 (“When another part of the computer system 10 needs to write data to the hard disk, it will flush the memory buffer that contains the audit markers to the physical disk.”). To flush or move the memory buffer’s data, the system must identify the presence of the data in the memory buffer; absent such identification the system would neither know what it is moving nor when the move has completed. A POSITA would have understood that any one of these events—the memory buffer being full, the processing unit requiring memory in the memory buffer, the computer system needing to write data to the hard disk—constitutes a “first event,” as claimed, that triggers identifying presence of the data in the memory buffer. EX1003, ¶66; EX1001, 5:46-49.

Long further teaches creating other information that is used to identify the presence of the archived data in the memory buffer responsive to the occurrence of the events discussed above. For example, it creates a marker in the memory buffer during “[s]torage of the data in the memory buffer.” EX1004, 7:59-8:13. The “audit marker 32 comprises a marker header 34 and a marker trailer 36” where captured data “is appended to the marker header 34.” *Id.*, 5:30-34; *id.*, FIG. 3.



Id., FIG. 3. EX1003, ¶67.

Long teaches using the marker trailer and marker header to determine the location of the marker. For example, “[t]he marker trailer 36 is used to access the audit buffer 30 in reverse order,” and “contains ... information required to determine the location of the marker header.” *Id.*, 5:54-59. Furthermore, the audit header stores information regarding the markers, and the audit header is specifically updated with information regarding the last marker (“LastMarker”) which indicates that new archive data has been placed in the audit buffer. *Id.*, 5:22-24 (Table 1, “LastMarkerLoc”); EX1003, ¶68.

Additionally, “all data is written out to the physical disk before any of the actual data is,” which “is done by having the memory buffer semaphored.” *Id.*, 8:18-20. A semaphore on the memory buffer indicates the presence of the data within the memory buffer to other processes. EX1019, 354 (defining “semaphore”). A POSITA therefore understood that the presence of the data in the memory buffer is identified

in response to a process checking that semaphore (before writing to physical disk), i.e., a “first event” as discussed above. EX1003, ¶69.

When implemented for a file-based system such as Kenley’s, these operations taught by Long result in the audit subsystem identifying the presence of the copied/archive file in the memory buffer (temporary storage location) responsive to a first event. *Supra* Section VIII.C. This identification would lead to the archive file being moved, or “flush[ed],” to the audit buffer on the secondary disk storage, where the archive file is stored. *E.g.*, EX1004, 8:20-23, 7:47-53; *see also* EX1014, 3:1-4 (“A file storage element in the secondary storage maintains data in the form of files”); EX1003, ¶70.

Thus, Kenley-Long renders obvious “the step of identifying presence of the archive file in the temporary storage location responsive to the occurrence of a first event.”

[1.5] (d) the step of transmitting the archive file to a second storage location responsive to a second event, the second storage location being an intermediate or a permanent storage location;

Long teaches transmitting the copied data to a second storage location responsive to a second event. For example, it teaches “archiving data to removable media” or “archive media.” EX1004, 7:1-3; *id.*, 7:10-14; *id.*, 8:29-9:9; *id.*, FIG. 6. This archiving to the removable storage media is responsive to a second event as it occurs “at predetermined intervals (for example, every five seconds),” based on a

“five second ticker.” EX1004, 8:36-42. Each completion of the ticker is an event, and a POSITA would have naturally implemented the ticker to alert the system after the passage of 5 seconds. Long discloses other events that trigger the writing of data to the archive media, e.g., detecting that the archive media is mounted with available storage space. *Id.*, FIG. 6, 8:52-59, 7:18-26. For purposes of claim 1, the removable media storage is a permanent storage because it is a persistent “archive media for external storage.” *Id.*, 8:33. Long FIG. 6 provides an exemplary data archive process, with 66 showing the data read from the audit buffer and 68 showing the data written to the archive/removable media. *See also* EX1004, 8:63-9:7 (describing reading data from the audit buffer and writing it to the archive media). In reading the data from the audit buffer and writing it to the archive storage, Long teaches transmitting the data to a second storage location. EX1003, ¶72.

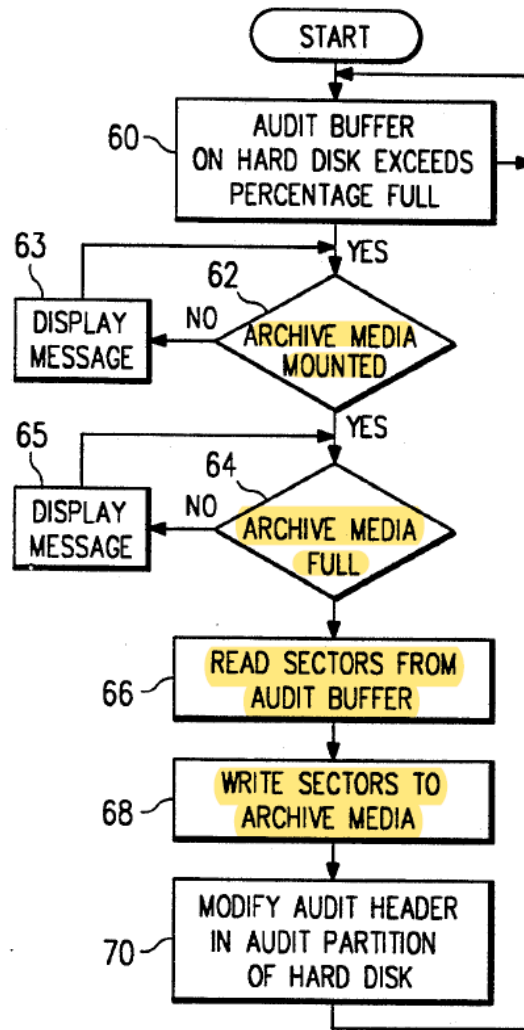


FIG. 6

EX1004, FIG. 6 (highlighting added); EX1003, ¶73 (explaining Long FIG. 6).

When implemented for a file-based system such as Kenley's, these operations taught by Long result in the transmitting of the copied/archive file to the "backing store" taught Kenley, respectively. *E.g.*, EX1014, 3:29-30 ("the backing store stores data organized as files"); EX1003, ¶74; *supra* Section VIII.C.

Kenley-Long thus renders obvious “the step of transmitting the archive file to a second storage location responsive to a second event, the second storage location being an intermediate or a permanent storage location.” EX1003, ¶75.

[1.6] (e) after storing the archive file in the temporary storage location, updating a database to indicate that the archive file is located in the temporary storage location;

Kenley-Long renders this step obvious. Kenley teaches a relational database that holds a “system catalog” for storing and maintaining the locations of backup copies of a file after the backup copy is moved (migrated). *E.g.*, EX1014, 3:17-28 (describing using a system catalog element and “maintaining” it after migrating files to the backing store), 6:35-40 (disclosing relational database storing “backup catalogs (i.e., lists of all files copied to specific backup savesets)”), 8:26-29 (describing recording to the “database 14 a catalog of all files backed up...”); *see also id.*, 2:6-14 (identifying not tracking the location of backup copies as a problem to be solved); EX1003, ¶76.

Long, in turn, teaches an audit header that is updated to reflect changes in the data’s storage location, including when the copied data is stored within the memory buffer (temporary storage location). EX1004, 7:49-53. The audit header “contains information that is used to manage the audit buffer 30, archive media ... and the general environment of the audit subsystem.” *Id.*, 4:44-47; *id.*, 4:57-60, 2:5-8. The audit header includes a number of fields that specify the location of data managed

by the audit subsystem, whether that data is stored in the memory buffer, audit buffer, or archive media. *Id.*, 4:61-5:24 (Table 1). For example, the audit header includes a “CacheLoc” field that provides the “current location of the audit buffer in the disk cache,” a “BufLoc” field that provides the “physical location of the beginning of the audit buffer on disk,” and a “CLoc” field that provides the “physical location of the last data in the buffer.” *Id.* Long discloses “construct[ing] the audit header” for a marker after “[s]torage of the data in the memory buffer” in the marker. *Id.*, 7:59-68. Then, when the data is moved from the memory buffer (temporary storage) to the audit buffer, the audit header is updated “to reflect the changes.” *Id.*, 7:47-53. Constant updating of this audit header to indicate the location of the data in the memory buffer and other locations, after storing the data in the respective locations, facilitates the audit subsystem’s accurate retrieval and movement of data for both backup and recovery. EX1003, ¶77.

TABLE 1	
Audit Header Fields	
Data Label	Data Description
Ver	The version of the audit system that created the audit partition.
Name	A name to assist in determining that the audit partition is valid.
HdrSeqNum	Sequence number that is used every time the header is written to disk.
Time	The time the audit partition was created or reset (cleared out).
CloseErr	The last audit error. This is used to determining if a failure occurred and the system came down before the audit

TABLE 1-continued

Data Label	Data Description	Audit Header Fields
	header was updated on the hard disk(s).	
SeqNum	The next Archive disk sequence number to use.	
QSeqNum	The last archive disk that contains a user quiescence marker.	
Flush	A note to the system that the audit header needs to be written to disk at the next available opportunity (used in memory only).	
LabelLoc	The physical location of the label on disk (audit header).	
BufLoc	The physical location of the beginning of the audit buffer on disk.	
CacheLoc	The current location of the audit buffer in the disk cache.	
CLoc	The physical location of the last data in the buffer. The next update of the buffer will begin at this address. This is on a sector by sector basis.	
CLocBytes	Current offset in the current sector that we are at (see CLoc).	
MaxSectors	The number of sectors that make up the audit buffer. This does not include the first sector that contains the audit header.	
MaxUsable	When the audit buffer gets to this point the archive system will start archiving.	
MinUsable	When the audit buffer gets to this point the archive system will stop archiving.	
UsedSectors	The number of sectors that have been used in the audit buffer.	
CurSector	The logical sector number currently in use by the audit buffer.	
FirstMarker	The contents of the first marker header.	
FirstMarkerLoc	The location of the first marker in the audit buffer.	
FirstMarkerOffset	The offset in the sector that contains the first marker.	
FirstMarkerSector	The logical sector that contains the first marker.	
LastMarker	The contents of the last marker header.	
LastMarkerLoc	The last marker location in the audit buffer.	
LastMarkerOffset	The offset in the sector that contains the last marker.	

EX1004, 4:61-5:24 (Table 1) (highlighting added).

Thus, in view of these complementary teachings, a POSITA would have been motivated with a reasonable expectation of success to implement Kenley-Long such that information on current locations of different backup files, including the audit header information taught by Long, was stored in the relational database catalog taught by Kenley. The audit header of Long, with all of its fields, already is nothing more than a database of information and thus a POSITA would have easily implemented the Kenley-Long system so as to store Long's audit header information in Kenley's relational database. EX1003, ¶78. The '348 patent itself only describes a "database" functionally as a place to record information "indicat[ing] the location of the archive files." *E.g.*, EX1001, 6:29-38.

And as with other elements, when implemented for a file-based system such as Kenley's, these database updates would indicate that the copied file (archive file) is located in the memory buffer (temporary storage location). EX1003, ¶79.

Kenley-Long thus renders obvious "after storing the archive file in the temporary storage location, updating a database to indicate that the archive file is located in the temporary storage location." EX1003, ¶80.

[1.7] (f) determining a final destination for the archive file;

The '348 patent provides no explanation for the term "final destination," nor is the recited "final destination" recited elsewhere in the claims. EX1003, ¶81.

Kenley teaches to predetermine the location of the data in its backing store archive storage, as it specifies moving "data contained in the selected data files to a predetermined staging volume of the backing store element." EX1014, 3:9-12; EX1003, ¶82.

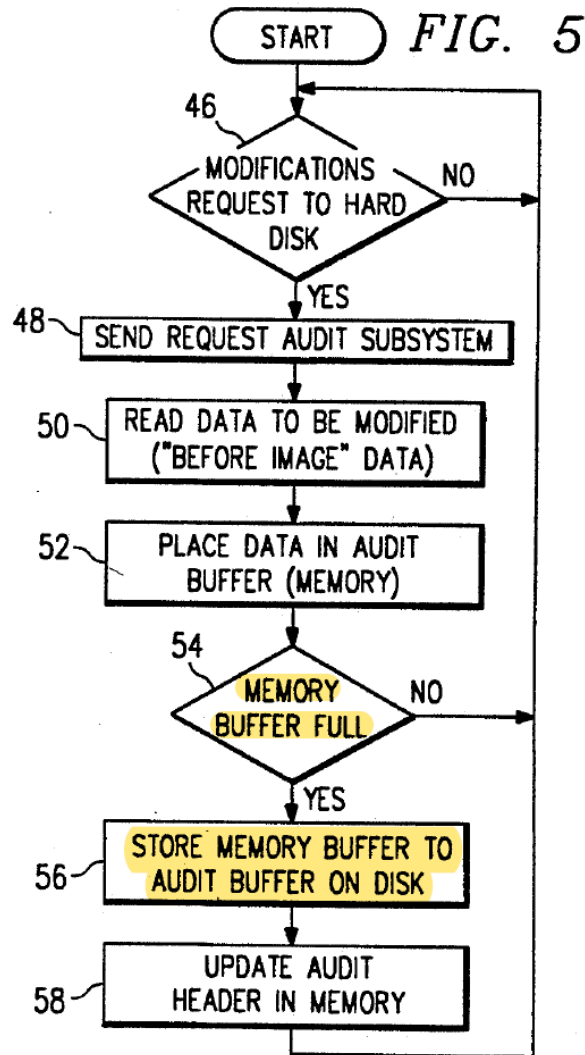
Similarly, as explained for Element [1.4], Long teaches "archiving data to removable media." EX1004, 7:1-14, 8:29-9:9, FIG. 6. Before storing the data to the removable media, Long first "determine[s] whether archive media is mounted" and "whether the archive media is full." *Id.*, 8:50-57. Archiving of the audit buffer begins once "a new media is mounted which is not full." *Id.*, 8:57-59. Long further discloses preparing an "archive track" to be written to the archive media, including an "archive end" marker at the end of the data to indicate the end of the archive track. *Id.*, 9:3-6;

id., 8:38-42 (“the size of the track is dependent upon the structure of the archive media”). Fields in the audit header specify an “Archive disk sequence number to use.” EX1004, 4:61-5:24 (Table 1) (describing a “SeqNum” as “[t]he next Archive disk sequence number to use” and “QSeqNum” as “[t]he last archive disk that contains a user quiescence marker”). Long thus teaches determining a final destination within the archive media to store the archive data, in order to prepare the “archive track.” EX1003, ¶83.

Thus, a POSITA would have implemented Kenley-Long to “determine[e] a final destination for the archive file,” for each archived file, at least so that the system would know where in secondary storage to move the archive file during the archive process. EX1003, ¶84.

[1.8] (g) moving the archive file from the temporary storage location to an intermediate storage location;

Long teaches moving the copied data from the memory buffer (temporary storage) to the audit buffer (intermediate storage location) prior to ultimately archiving the data to the removable media. EX1004, 7:47-53. For example, Long’s audit subsystem “writes the data stored in the memory buffer to the end of the audit buffer 30 on disk” when the memory buffer is full or when the memory buffer is needed for another purpose. EX1004, 7:47-53; *id.*, 8:20-23 (“[F]lush the memory buffer that contains the audit markers to the physical disk.”); *id.*, FIG. 5.



EX1004, FIG. 5 (highlighting added). EX1003, ¶85.

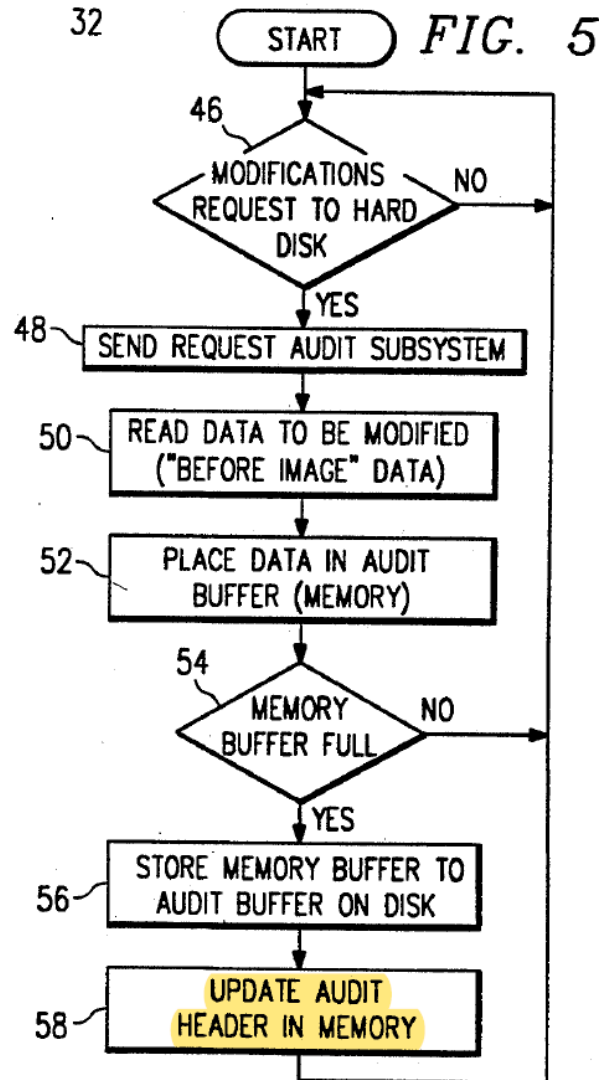
The audit buffer is an intermediate storage location as taught by Long, at least because the data is stored in the audit buffer in between being stored in the memory buffer (temporary storage) and the archive media (second/permanent storage location). EX1004, 8:29-9:9 (explaining the archive process); *id.*, FIG. 6; EX1003, ¶86.

When implemented for a file-based system such as Kenley's, this would involve movement of the copied file (archive file) to an audit buffer as taught by Long, in a hard disk secondary storage such as that taught by Kenley. EX1003, ¶87; EX1014, 3:1-4 (“A file storage element in the secondary storage maintains data in the form of files”), 4:56-60 (secondary storage as hard disk).

Kenley-Long thus renders obvious “moving the archive file from the temporary storage location to an intermediate storage location.” EX1003, ¶88.

[1.9] (h) updating the database to indicate the archive file is located in the intermediate storage location; and

Long teaches updating its audit header to indicate the copied data (archive data) is located in the audit buffer (intermediate storage): after “writ[ing] the data stored in the memory buffer to the end of the audit buffer 30 on disk” Long specifies “updat[ing] the audit header (in the memory buffer) to reflect the changes.” EX1004, 7:47-53; *id.*, FIG. 6.



EX1004, FIG. 5 (highlighting added). The audit header is updated to indicate “[t]he physical location of the beginning of the audit buffer on disk.” EX1004, 4:61-5:24 (Table 1) (defining “BufLoc”). EX1003, ¶89.

As discussed above for Element [1.6], Kenley teaches a complementary relational database with a backup catalog containing, e.g., “lists of all files copied to specific backup savesets.” EX1014, 6:35-40. For the same reasons explained above, a POSITA would have naturally implemented Kenley-Long such that a backup

catalog database, as taught by Kenley, was updated with the location of the copied file (archive file) in the secondary storage audit buffer, upon movement of the copied file to that secondary storage. *Supra* Section VIII.D.1 ([1.6]); EX1003, ¶90.

Kenley-Long thus renders obvious “updating the database to indicate the archive file is located in the intermediate storage location.” EX1003, ¶91.

[1.10] (i) after moving the archive file to the second storage location, updating the database to indicate the archive file is located in the second storage location.

Long teaches updating its audit header to indicate that the copied/archived data is location in the removable archive media (second storage location), after the data has been moved to that location. Long teaches, “[a]fter the write [to the archive media] is completed, the audit header in memory is modified to show the removal of the audit markers from the audit buffer 30.” EX1004, 9:7-9; *id.*, FIG. 6.

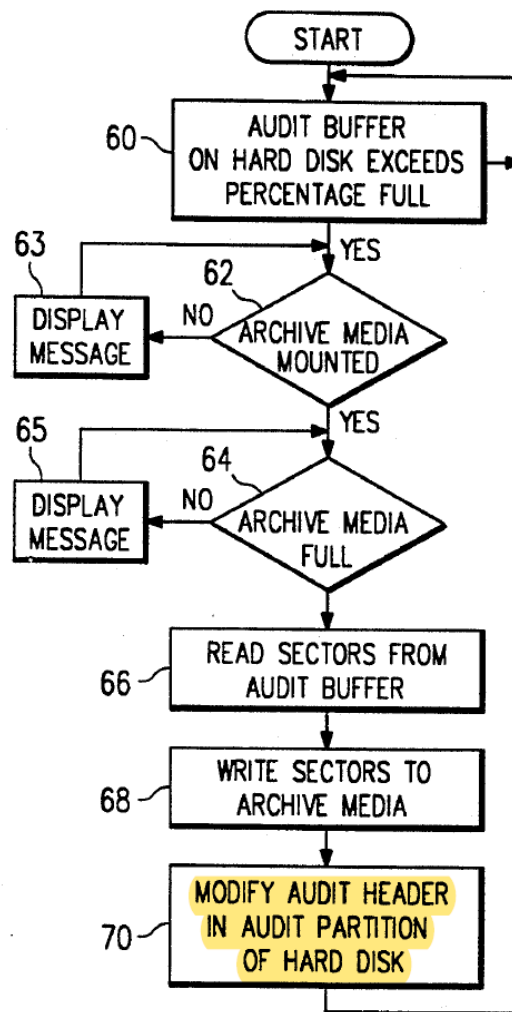


FIG. 6

EX1004, FIG. 6 (highlighting added). EX1003, ¶92.

The audit header includes fields specifying the location of data in the archive media that are “used to manage the ... archive media,” such as the “Archive disk sequence number to use” and the “last archive disk that contains a user quiescence marker.” *Id.*, 4:44-47; *id.*, 4:61-5:24 (Table 1) (describing a “SeqNum” as “[t]he next Archive disk sequence number to use” and “QSeqNum” as “[t]he last archive disk that contains a user quiescence marker”). A POSITA would have understood the

audit header as a database indicating data locations. *Supra* Section VIII.D.1 ([1.6]); EX1003, ¶93.

As discussed above for Elements [1.6] and [1.9], Kenley teaches a database with a backup catalog containing, e.g., “lists of all files copied to specific backup savesets.” EX1014, 6:35-40. For the same reasons explained above, a POSITA would have naturally implemented Kenley-Long such that a backup catalog database, as taught by Kenley, was updated with the location of the copied file (archive file) in the removable media archive or backing store (second / permanent storage location), upon movement of the copied file to removable media. *Supra* Section VIII.D.1 ([1.6]); EX1003, ¶94.

Kenley-Long therefore renders obvious, “after moving the archive file to the second storage location, updating the database to indicate the archive file is located in the second storage location.” EX1003, ¶95.

2. Claim 2

[2] The method of claim 1, wherein the second storage location includes a personal attached storage device.

The '348 patent defines “Personal Attached Storage Device” as “any internal or external storage device connected to a computing device.” EX1001, 4:32-33. In the Kenley-Long system, it would have been an obvious design choice to have the backing store (second storage location) use one of the many storage devices taught

by Kenley and Long, each of which is either an internal or external storage connected to the computing device, e.g.,:

- backing store devices taught by Kenley, such as “erasable optical, write-once-read-many (WORM) disks, or tape volumes” (EX1014, 4:61-66), or
- the removable “archive media for external storage” taught by Long (EX1004, 8:32-33; *id.*, 7:12-14 (“removable media, such as a floppy disk”), 2:46-50 (disclosing a computer including “floppy drives 16 (or other removable media device)”), FIG. 1).

EX1003, ¶¶96-97.

3. Claim 3

[3] The method of claim 1, wherein the second storage location includes a network attached storage device.

In implementing the Kenley-Long system, it would have been at least obvious to include a network attached storage device as the archive/backing store. EX1001, 4:34-40 (broadly defining “Network Attached Storage Device” as including “any storage device connected directly to a network to which a first computing device is also temporarily or permanently connected”). EX1003, ¶¶98-99.

Kenley, as background, notes “particular interest” in “the Network File System (NFS) protocol, developed by Sun Microsystems, allowing users to share files across a local area network such as Ethernet.” EX1014, 1:30-33; *see also id.*,

1:42-45 (noting “data storage requirements of professional work groups employing networked workstations”), 2:11-14 (discussing “migrating data ... so that the files can be read and utilized by computers or processors in the network”). Kenley then teaches use of the “conventional NFS network” to provide “rapid and transparent data migration” by “sending conventional NFS requests and responses to the network.” *Id.*, 13:42-50. It thus would have been at least obvious to implement Kenley-Long such that it used the “conventional NFS network” to connect the personal computer / computing device to the devices providing the temporary and intermediate storage locations. *E.g., id.*, 2:11-14 (presenting one problem to solve as “transferring computer data files ... so that the files can be read and utilized by computers or processors in the network”); *id.*, 6:29-66, FIGS. 1A-B & 2. EX1003, ¶99.

Thus, it was obvious to implement Kenley-Long such that the backing store (second storage location) included a network attached storage device, e.g., connected via an Ethernet with files accessible using the conventional network file sharing (“NFS”) protocol, as taught by Kenley. EX1003, ¶100-101.

4. Claim 4

[4] The method of claim 1, wherein the second storage location includes a peer-to-peer storage device.

The '348 patent defines “Peer-to-Peer Storage Device” as “any storage area (device, collection of devices, etc.) that can be accessed by the computing device when it is sharing resources with other network or Internet accessible computers.” EX1001, 4:45-48. EX1003, ¶102.

In implementing the Kenley-Long system, it would have been at least obvious to include a peer-to-peer storage device as the archive/backing store, for the same reasons provided above regarding Claim 3. Kenley explains that “[t]he data storage requirements of professional work groups employing networked workstations can measure billions of bytes—i.e., Gigabytes—in size,” and that “[a]dditional file servers can be incorporated into an existing computer network” to increase data storage capacity. EX1014, 1:42-51. Thus, it would have been obvious to implement Kenley-Long such that the second storage location includes a peer-to-peer storage device, because Kenley describes incorporating a file server (i.e., storage area) into an existing computer network that can be accessed by a computing device when it is sharing resources (i.e., data storage resources) with networked workstations (i.e., network accessible computers). EX1003, ¶¶103-104; *see also infra* Section VIII.D.5

(claim 5) (explaining obviousness of shared internet-based external storage device for the archive).

5. Claim 5

[5] The method of claim 1, wherein the second storage location includes an internet storage area network.

The '348 patent defines “Internet storage area network” as “any storage area (device, collection of devices, etc.) that can be accessed by the computing device when the computing device is temporarily or permanently connected to the Internet.” EX1001, 4:41-44. EX1003, ¶105.

As early as 1993, it was well known that the Internet was “commonly used to transfer ... files between computers.” EX1018, 3:13-19 (background provided in patent issued from application filed November 4, 1993). Even then, the Internet was a known alternative to the local area network (“LAN”), e.g., Ethernet, taught by Kenley: “The Internet is similar to a LAN/WAN in that it was designed to allow many computers of differing types to interconnect and exchange information and programs.” EX1018, 3:37-39; *see also* EX1005, 6:9-36 (1999 patent describing remote backup storage where “logical connections to one or more remote computers,” such as a “memory storage device,” included “commonplace” LAN and WAN connections, “such as the Internet”). EX1003, ¶106.

It was also well known to access external drives using the Internet, as “[d]rives on a remote system defined using NFS [we]re available for mounting by other remote systems on the Internet.” EX1018, 3:64-66; EX1005, 6:9-36; *see also supra* Section VIII.D.3. By 2000, people had been “sharing data on the Internet for decades already” (EX1007, 544) and it was known that “[o]ne of the earliest Internet-based storage applications to arise since the emergence of the World Wide Web is system backup and recovery” (*id.*, 556). Thus, the POSITA knew how to implement Internet-connected drives, and that they were an established design choice for backup/recovery storage. EX1003, ¶107; *supra* Section IV.

Given this knowledge, and Kenley’s teachings to use the NFS protocol for transmission of files to networked storage devices (*e.g.*, EX1014, 1:30-33, 13:44-48), a POSITA would have naturally implemented the archive/backing store of Kenley-Long using a removable media (*e.g.*, optical disk) accessed via a wide-area network such as the Internet. At the very least, this was an obvious, well-established design choice. Moreover, the POSITA would have been motivated to do so for at least two benefits. First, to reduce chances of the backup being destroyed. EX1007, 556 (“[T]he chances are pretty good that the web site will not be destroyed by the same disaster that strikes a system in the office or at home.”). Second, to beneficially conserve local storage space and allow computers in different locations to access and share external storage resources. EX1018, 13:28-31; *see also id.*, 3:64-66. A

POSITA would easily implement this known alternative using the NFS protocol taught by Kenley. *E.g.*, EX1018, 3:61-66; *see also id.*, 34:46-65; EX1003, ¶108.

The obviousness of this design choice is confirmed by the '348 patent itself, which states that the storage device “may be a personal attached storage device, a network attached storage device, an Internet storage area network, a peer-to-peer storage device, or other storage device,” without any explanation of how to implement any of these alternatives or why one would be preferred over the other. EX1001, 5:32-35. It treats them as interchangeable and easily implemented because they were. Accordingly, claim 5 was obvious over Long, Kenley, and the knowledge of a POSITA. EX1003, ¶¶109-110.

6. Claim 6

[6] An article of manufacture comprising a computer usable medium having computer readable program code for performing the method of claim 1.

Kenley-Long renders this obvious. Kenley and Long each teach software programs for implementing backup-archival processes. *E.g.*, EX1014, 1:6-9; EX1004, 6:60-63. Long further discloses “processing circuitry” performing its backup-archival process. *E.g.*, *id.*, 1:67-2:2, 2:45-57, FIGS. 1, 2a, 10:10-11:13 (claims 1-15) (claiming computer system comprising processing circuitry to perform the audit/archive method).

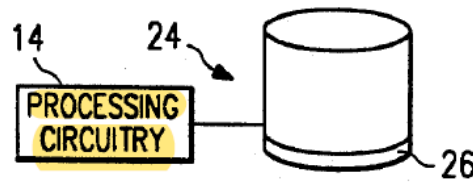


FIG. 2a

EX1004, FIG. 2a (highlighting added). EX1003, ¶111.

Given this, and that the claim 1 method is obvious over Kenley-Long (*supra* Section VIII.D.1), Kenley-Long renders claim 6 obvious. EX1003, ¶112.

7. Claim 8

Elements [8.1]-[8.6] are identical to elements [1.1]-[1.6], and are satisfied by Kenley-Long for the same reasons as explained above for elements [1.1]-[1.6]. *See supra* Section VIII.D.1; EX1003, ¶¶113-114.

[8.7] (f) the step of moving the archive file from the temporary storage location to an output buffer;

[8.8] (g) the step of updating the database to indicate that the archive file is located in the output buffer;

Kenley-Long satisfies these elements, as it renders obvious moving the copied file (archive file) from the memory buffer (temporary storage) to the audit buffer (output buffer), and then updating the database catalog to indicate the location of the copied file in the audit buffer. *See supra* Section VIII.D.1 ([1.8-1.9]); EX1003, ¶115.

[8.9] (h) the step of transmitting the archive file from the output buffer to the intermediate or the permanent storage location; and

Kenley-Long satisfies this element, as it renders obvious transmitting the copied/archive file from the audit buffer (output buffer) to the backing store/archive (permanent storage location). *Supra* Section VIII.D.1 ([1.5]); *see also* EX1004, 7:1-3, 7:10-14, 8:29-9:9, FIG. 6. As the archive media (e.g., backing store) is separate from the audit buffer, the copied file must be transmitted to the archive media. EX1003, ¶116.

[8.10] (i) the step of updating the database to indicate that the archive file is located in the intermediate or the permanent storage location.

Kenley-Long renders this element obvious. *Supra* Section VIII.D.1 ([1.10]); EX1003, ¶117.

8. Claims 9-11

Claims 9-11 are identical to claims 2-3 and 5, respectively, and are obvious over Kenley-Long for the same reasons explained above for claims 2-3 and 5. *Supra* Sections VIII.D.2, VIII.D.3, VIII.D.5; EX1003, ¶118; EX1023, 2.

9. Claims 12-15

Elements [12.1]-[12.5], [13.1]-[13.5], [14.1]-[14.5], and [15.1]-[15.5] are identical to one another and to Elements [1.1]-[1.5], and are satisfied by Kenley-Long for the same reasons explained above for Elements [1.1]-[1.5]. *Supra* Section VIII.D.1; EX1003, ¶¶119-120; EX1023, 1.

a) **Element [12.6]**

[12.6] wherein the second event includes a message indicating when the second storage location is available.

Long teaches “determin[ing] whether archive media is mounted” and then “whether the archive media is full” prior to transmitting the audit data from the audit buffer to the archive media. EX1004, 8:50-59; FIG. 6; *see also* EX1014, 5:36-39, 5:57-63, 6:40-43 (describing managing access to backing store volumes), 16:38-59 (“No I/O processes can occur until the volume is physically mounted in a drive,” thus data migration is blocked while the volume is mounted). These determinations would have required communication between the audit subsystem and the removable archive media device, and thus it would have been at least obvious to implement Kenley-Long using messages sent to the audit subsystem, indicating the availability of the removable device. EX1003, ¶121.

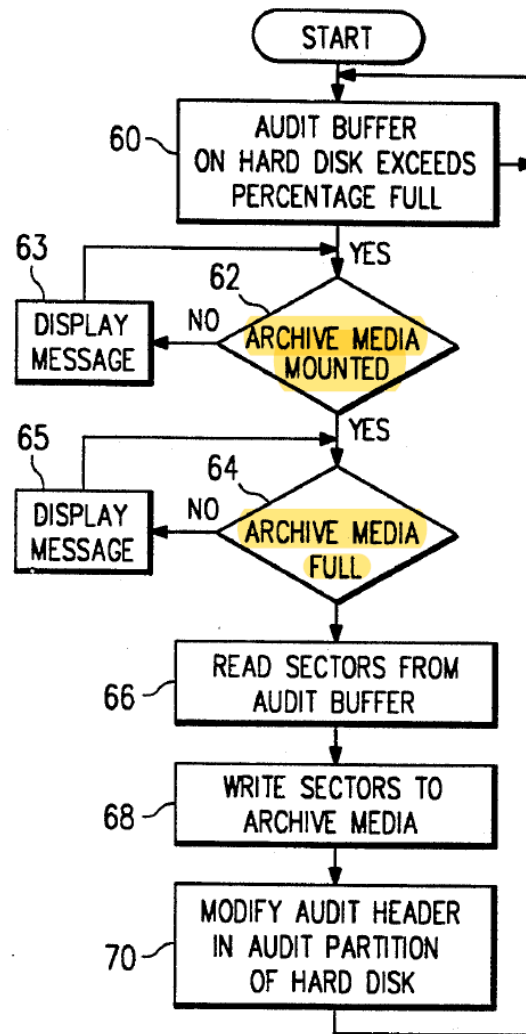


FIG. 6

EX1004, FIG. 6 (highlighting added).

b) **Element [13.6]**

[13.6] wherein the second event includes a message from a timer.

Long teaches that the second event includes a message from a timer. *Supra* Section VIII.D.1 ([1.5]). It would have been obvious to implement Kenley-Long such that the “five second ticker” taught by Long (EX1004, 8:36-37) generated a

message to signal completion of a five second (or other) time increment and trigger transmission of the file to the archive media. EX1004, 8:38-42; EX1003, ¶122.

c) **Element [14.6]**

[14.6] wherein the first event includes a message from a timer.

Kenley-Long renders this element obvious. *Supra* Section VIII.D.1 ([1.5]). Long teaches a timer for archiving data from the audit buffer. EX1004, 8:36-42 (“five second ticker”). Long also teaches flushing the memory buffer to the audit buffer on disk “[w]hen the memory buffer is full ... or the processing unit 14 requires memory used in the memory buffer for another purpose.” EX1004, 7:47-53. A POSITA would have also found it obvious to implement a “ticker” (i.e., timer) for triggering transfer of the audit data in the memory buffer to the audit buffer, in order to further “increase the throughput of the audit subsystem” as doing so would allow the data transfer “to take place in the background” and thus be “asynchronous to the rest of the system.” EX1004, 7:43-46; *id.*, 42-45. The express motivation for using the timer to trigger the archival transfer also applies to the memory buffer. The POSITA would have easily implemented such a timer, with a reasonable expectation of success, as reflected in Long’s teachings to use such a timer for similar data transfers. The ease of implementation is further confirmed by the ’348 patent’s failure to provide any meaningful description of how to implement such a timer. Other background prior art further confirms the obviousness of the claimed timer.

See, e.g., EX1005, 10:53-55 (“the triggering event ... would be the expiration of an elapsed time rather than the accumulation of an amount of data”); *id.*, 11:15-16; EX1003, ¶123.

d) Element [15.6]

[15.6] wherein the first event is different from the second event.

Long teaches that the events causing movement of data from the memory buffer (temporary storage) to the audit buffer (intermediate storage) are distinct from the events causing movement of data from the audit buffer to the archive media (permanent storage). *Compare* EX1004, FIG. 5 *with id.*, FIG. 6; *see also supra* Section VIII.D.1 ([1.4]-[1.5]). Long teaches several types of “first event” that lead to moving data from the memory buffer (e.g., the memory buffer being full, the processing unit requiring memory). *Supra* Section VIII.D.1 ([1.4]). It teaches distinct types of “second event” that lead to movement of data from the audit buffer (e.g., the audit buffer reaches a certain capacity or a timer). EX1004, 8:38-42; *supra* Section VIII.D.1 ([1.5]). Thus, it was obvious to implement Kenley-Long such that the first event is different from the second event. EX1003, ¶124.

10. Claim 16

[16] The method of claim 15, wherein the resident program is separate from an operating system.

In Kenley-Long, modification requests (instructions for performing an operation on an operating file) originate from an application or the operating system, and are received by the audit subsystem. *Supra* Section VIII.C, VIII.D.1 ([1.2]). When the request originates from an application and is received by the audit subsystem, the resident program is separate from the operating system. EX1003, ¶125.

11. Claim 17

[17] The method of claim 16, where the resident program comprises a program having control over file operations on one or more operating files.

In Kenley-Long, an application controls file operations on at least one operating file as it is able to change, e.g., “write” data. EX1004, 4:23-25, 7:28-29, 7:40-42; *supra* Section VIII.D.1 ([1.2]). Additionally, the audit subsystem also has control over file operations, as it is able to read/write data from/to memory, the local hard disk, and archive media. *E.g.*, EX1004, 7:28-64, FIGS. 5-6. Thus, Kenley-Long renders claim 17 obvious. EX1003, ¶126.

12. Claim 18

[18] The method of claim 15, wherein the operation comprises one of read, write, save, rename, delete, copy, move, open, close, or create operations.

Kenley-Long renders this claim obvious. *Supra* Section VIII9 (claim 15). Kenley discloses “read” and “write” access to a file, “open/close” operations, and “file delete and rename operations.” EX1014, 11:29-32, 17:23-25 (discussing monitoring “all open/close and read/write operations”). Furthermore, a POSITA would have understood that the ability of the “application to modify the contents of the hard disk” as taught by Long indicates that the operation comprises at least one of read, write, save, rename, delete, copy, move, open, close, or create operations, insofar as these operations modify the contents of the hard disk. EX1004, 7:28-29; EX1003, ¶127.

13. Claim 19

[19] The method of claim 15, wherein the operating file comprises one of a system file or a user file.

Kenley teaches backing up and archiving both system and user files. EX1014, 2:6-10 (discussing “users” and “their own information files”), 3:29-32 (discussing “enabling a user or an applications program to request selected files stored in backing store”). Long teaches archiving both system and user files. EX1004, 4:23-25 (“[T]he data on the hard disk is frequently changed during operation of an application program or by the operating system.”); *id.*, 7:28-29; *see also* EX1014, 1:30-33, 1:45-47. Thus, it was obvious to implement Kenley-Long to archive both system files and user files. EX1003, ¶128.

14. Claim 20

[20] The method of claim 15, wherein the archive file comprises at least some data of the operating file.

Long teaches the “before image” data (i.e., the archive file) is “[t]he data to be modified” and therefore comprises at least some data of the operating file, as claimed. EX1004, 7:37-40; EX1003, ¶129; *supra* Section VIII.C.

15. Claim 23

[23] The method of claim 15, wherein the temporary storage location is located in a storage device internal to the computing device.

Long teaches a “semiconductor memory buffer” internal to the processing unit 14 of the computer system 10, thus at least rendering obvious temporary storage in a storage device internal to the computing device. EX1004, 7:43-46; *id.*, 2:45-50; *see also* EX1014, 6:3-4 (“system memory”), 6:64 (“core memory”); EX1003, ¶130.

16. Claim 24

[24] The method of claim 23, wherein the temporary storage location comprises a buffer located in the storage device.

Long teaches a memory buffer as its temporary storage location. EX1004, 7:43-46; *supra* Section VIII.D.1 ([1.3]-[1.4]); EX1003, ¶131.

17. Claim 25

[25] The method of claim 15, wherein the temporary storage location is located in a storage device external to the computing device.

Kenley-Long renders obvious an external storage device for the temporary storage location. Kenley discloses a file server with “a hierarchical storage structure” that enables “rapid and transparent data migration.” EX1014, 13:42-44, FIG. 2. The file server includes “RAM 30, magnetic disks 18, and optical disks 22.” *Id.*, 13:48-50. A “RAM buffer cache” allows for quickly accessing files. *Id.*, 14:59-68 (files are “also written asynchronously [from RAM] to magnetic disk”).

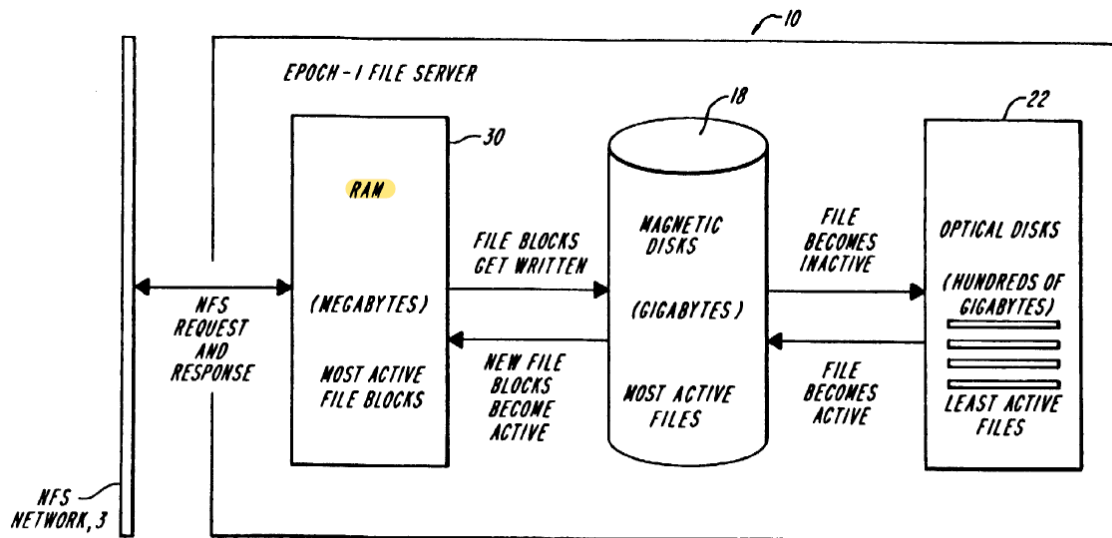


FIG. 2

EX1014, FIG. 2 (highlighting added).

It would have been obvious to implement Kenley-Long with a temporary storage location in an external storage device such as the RAM of the file server taught by Kenley, because doing so would reduce contention for storage internal to the computing device by offloading the backup-archive process to a file server accessible over a network. Further, the Kenley FIG. 2 external storage devices, and

others, were well-known and easily implemented by a POSITA. *Supra* Sections VIII.D.2, VIII.D.5 (claims 2, 5); *see also* EX1005, 5:48-57 (explaining various types of computer-readable media known to and usable by a POSITA), 6:43-57, 5:10-20; EX1003, ¶¶132-133.

18. Claim 26

[26] The method of claim 15, wherein the first event comprises one of the following: a message from a process that created the archive file from the operating file, a message from the resident program, or a message from a timer.

Long teaches various events that trigger identifying the presence of the data in the memory buffer (first events), including “[w]hen the memory buffer is full ... or the processing unit 14 requires memory used in the memory buffer for another purpose.” EX1004, 7:47-53; *supra* Section VIII.D.1 ([1.5]). It was also obvious to use a timer message as the first event. *Supra* Section VIII.D.9 ([14.6]). Thus, Kenley-Long renders obvious Claim 26. EX1003, ¶134.

19. Claim 27

[27] The method of claim 15, wherein the second event comprises one of the following: a message indicating when the intermediate or the permanent storage location is available, a message indicating when the intermediate or the permanent storage location is inserted, a message indicating high-bandwidth availability, a message from a server associated with the intermediate or the permanent storage location, or a message from a timer.

Kenley-Long renders claim 27 obvious, at least because Long teaches a message from a timer, a message indicating when the permanent storage location is

available, and a message indicating when the permanent storage location is inserted as second events triggering archiving. EX1004, 8:36-59; *supra* Sections VIII.D.1 ([1.5]), VIII.D.9 ([12.5]); EX1003, ¶135.

20. Claim 28

[28] The method of claim 15, wherein the intermediate or the permanent storage location is located in one of a personal attached storage device, a network attached storage device, a Internet storage area network, or a peer-to-peer storage device.

Kenley-Long renders obvious permanent storage locations that include those recited in claim 28. *Supra* Sections VIII.D.2-5 (claims 2-5).

21. Claim 29

[29] A non-transitory computer readable medium comprising program instruction for performing the method of claim 15.

Long and Kenley each describes implementing its technology as a program/software run by a computing system, using a processor. *E.g.*, EX1004, 1:67-2:2, 2:45-57, 6:60-63 (“memory-resident program.”), FIGS. 1-2a, 10:10-11:13 (claims 1-15); EX1014, 5:48-6:18, FIGS 1A-3B. Thus, Kenley-Long at least renders obvious implementing the archive system using program instructions on non-transitory computer readable media. *Supra* Section VIII.D.9 (claim 15); EX1003, ¶137.

22. Claim 30

[30] The method of claim 15, wherein the intermediate or the permanent storage location is accessible by the computing device through the Internet.

Kenley-Long renders this limitation obvious. *Supra* Section VIII.D.5 (claim 5); EX1003, ¶138.

23. Claim 31

[31] The method of claim 15, wherein the second storage location is the permanent storage location.

Kenley-Long satisfies this element. *Supra* Section VIII.D.9 ([15.5]); EX1003, ¶139.

IX. GROUND 2: CLAIMS 1-6, 8-20, AND 23-31 ARE OBVIOUS OVER CABRERA, SCHNEIDER, AND WHITING

A. Cabrera (EX1005)

Cabrera (EX1005) issued September 14, 1999, and thus qualifies as prior art under at least Sections 102(a-b). EX1005, Cover. Cabrera is analogous art to the '348 patent because both are within the same field of endeavor of preserving and managing data files. *Id.*, 1:7-12; EX1003, ¶¶140-141.

Cabrera discloses “systems and methods for transferring or archiving data from a local storage area to a remote storage area,” including “temporarily storing or staging data prior to its transfer to remote storage.” EX1005, 1:7-12; *id.*, 2:55-61. These backup/archive systems duplicate an entire file into a temporary storage area,

thereby “allowing access to files while archive or backup copies are created.” *Id.*, 1:51-54; *id.*, 1:64-2:10. EX1003, ¶142.

“[D]ata that is to be backed up or archived” is stored as a “data file” in a local storage area. *Id.*, 6:59-7:2, FIG. 2. The data file is copied from the local storage area to the temporary “staging storage area,” in response to an event. *Id.*, 7:3-46, FIG. 2; *see also id.*, 9:38-43, 9:43-10:36, FIG. 4.

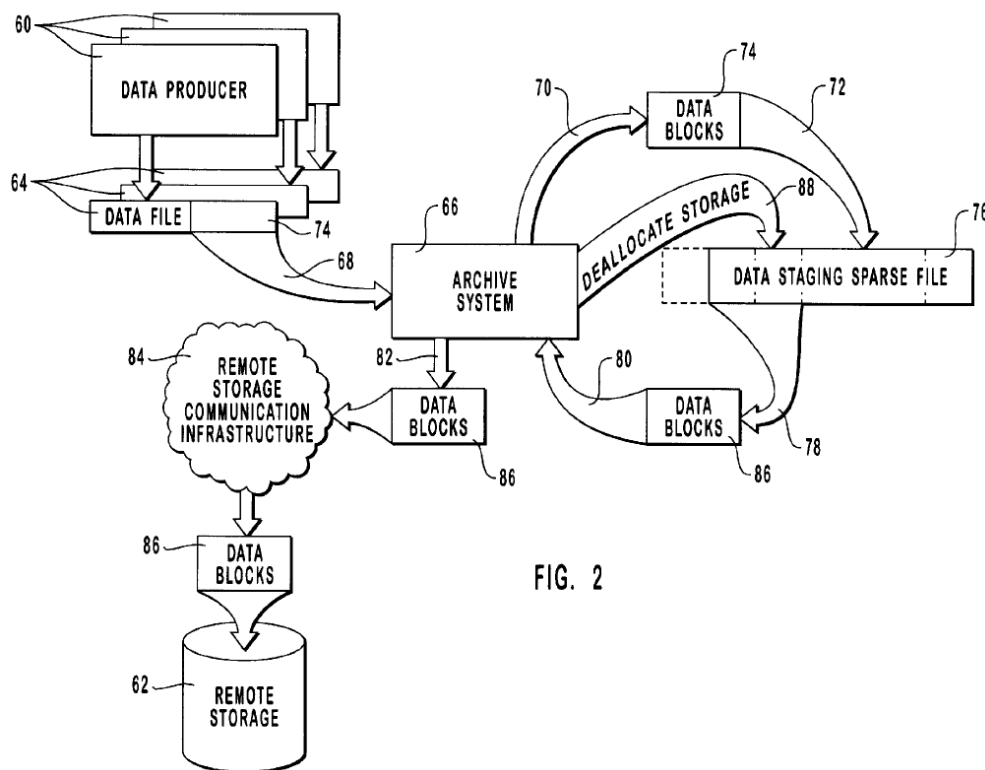
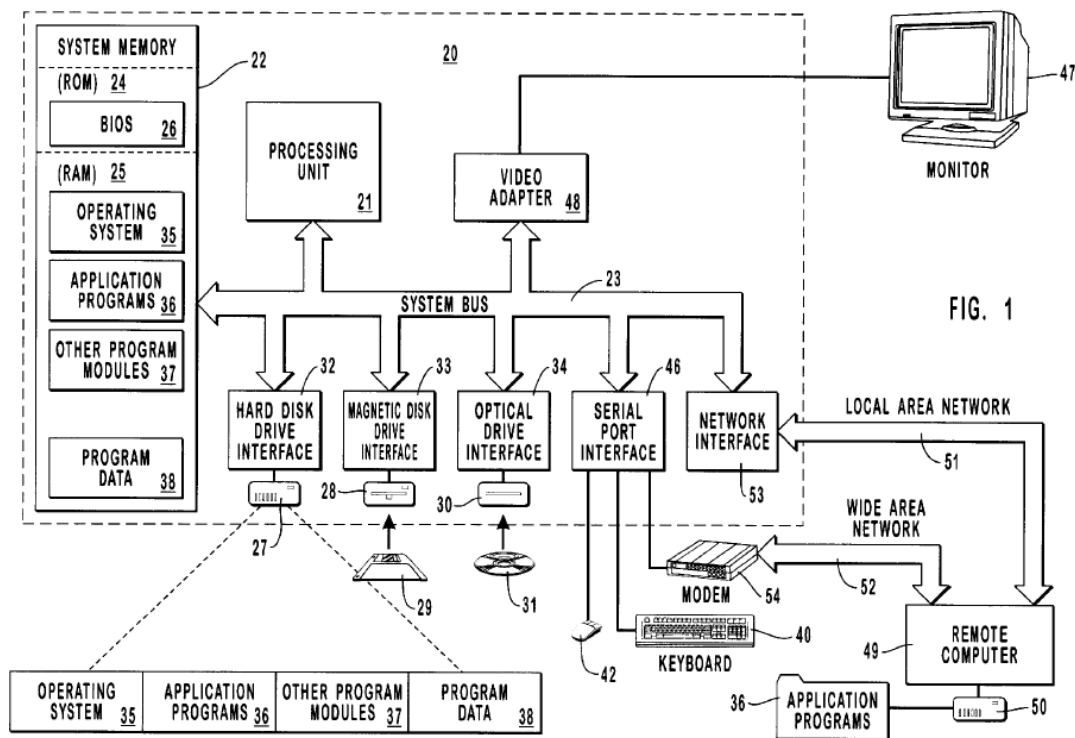


FIG. 2

Id., FIG. 2. EX1003, ¶143.

Cabrera contemplates archiving to both “remote” and “local” storage media. *Id.*, 3:3-7, 6:9-13. Local and remote connectivity includes “commonplace” local area networks (LANs) or wide area networks (WANs). *Id.*, 6:16-21; *id.*, FIG. 1.



Id., FIG. 1. EX1003, ¶144.

Files are copied from local storage to the staging area responsive to an event, e.g., “the expiration of a given time delay ... or any other triggering event.” *Id.*, 10:53-60; *see also id.*, 3:36-38; *id.*, 10:49-55, 11:10-27 (additional triggering events). Files are then “moved from the staging area to remote storage” (*id.*, 3:53-56), which may involve “an intermediate system or subsystem” (*id.*, 7:49-60). Cabrera describes an “event monitor ... to watch for desired triggering events and to initiate appropriate action when the events occur.” *Id.*, 12:62-13:13. EX1003, ¶¶145-146.

B. Schneider (EX1020)

Schneider (EX1020) published March 11, 1999, and thus qualifies as prior art under at least Sections 102(a-b). EX1020, Cover. Schneider is analogous art to the '348 patent because it is from the same field of endeavor of preserving and managing data files. EX1001, 1:12-16; EX1020, 1:15-17; EX1003, ¶¶147-148.

Schneider teaches maintaining the current version of a file on the user's local disk, while maintaining historic data on a separate "external disk" remotely accessible via a network. EX1020, 112:26-114:7. To maintain historic versions of a file, Schneider teaches "fundamental methods of recording the original state of information prior to its being altered," which includes "automatically saving the prior states of altered data on their hard disk" by "first reading data about to be overwritten and saving it in a disk-based history buffer." *Id.*, 34:12-13; *id.*, 34:29-35:4. EX1003, ¶149.

C. Whiting (EX1006)

Whiting (EX1006) issued July 7, 1998, and thus qualifies as prior art under at least Sections 102(a-b). EX1006. Whiting is analogous art to the '348 patent because it is from the same field of endeavor of preserving and managing files. EX1006, 1:5-8; EX1003, ¶¶150-151.

Whiting discloses a "system for backing up files from disk volumes on multiple nodes of a computer network to a common random-access backup storage

means.” *Id.*, Abstract; *id.*, 6:52-56 (describing migrating files through hierarchical storage). Whiting teaches a “backup directory” that “indicat[es] where the data for each [backup data] file is to be found.” EX1006, 8:46-50; *id.*, 6:23-27; *id.*, 8:41-13:9. A related “global directory database file” enables searching all backup files for all users. *Id.*, 20:6-38; *id.*, 20:6-26:36. The global directory database is updated to account for “new and updated files.” *Id.*, 25:46-53.

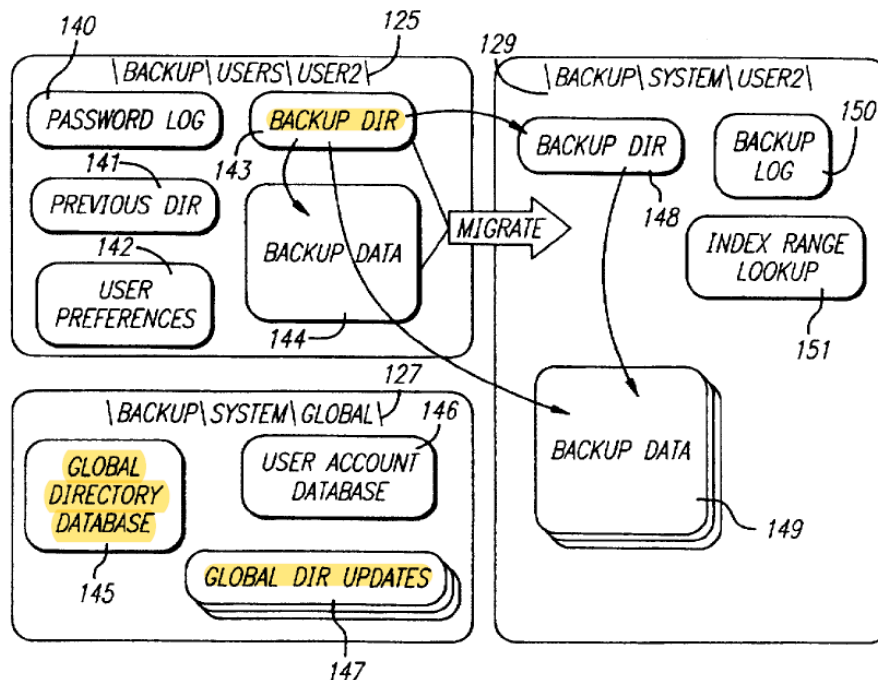


FIG. 3

Id., FIG. 3 (highlighting added). EX1003, ¶¶152-153.

D. Combining Cabrera, Schneider, And Whiting

As explained in the following paragraphs, a POSITA implementing Cabrera would have been motivated, with a reasonable expectation of success, to incorporate

Schneider's teachings of performing automatic backup responsive to file operations such as write operations.

Cabrera discloses that "any ... triggering event" may trigger the backup or archival process, such as "the expiration of a given time delay" or "a command to stage data." EX1005, 10:56-61. Meanwhile, Schneider notes a "shortcoming" of such approaches as potentially lacking a "recent backup," which could lead to loss of information. EX1020, 3:10-17. Schneider overcomes this shortcoming by "providing for a continuously running disk backup system." *Id.*, 3:10-17; *see also id.*, 12:1-8, 23:11-12, 24:21-23, 36:12-17. These benefits would have motivated the POSITA to use a file-manipulation event (e.g., save, write, or close event), as taught by Schneider, within the archive-backup system of Cabrera, in order to "eliminate the need to pause and make backups or decide which files should be backed up." EX1020, 5:24-25; EX1003, ¶154.

The POSITA had a reasonable expectation of success in implementing Schneider's operation-triggered archive process in the Cabrera system at least because POSITAs knew how to intercept file operations and use them to trigger other actions. *E.g.*, EX1020, 30:4-11 (teaching "How to Intercept Disk Activity"); EX1021, 4:2-7 (describing intercepting "calls" that occur "whenever a program wishes to access a file," in order to "record modifications to files"); *id.*, 20-25. Cabrera itself describes monitoring and capturing "changes as they occur" "to a disk

volume in order to allow recovery of the volume.” EX1005, 12:21-25. Thus, the POSITA would have easily modified Cabrera to detect instructions to operate on an operating file and use them as triggering events for its backup-archival process. EX1003, ¶155.

A POSITA further would have been motivated to incorporate the backup directory of Whiting to improve Cabrera’s ability to track “where the data for each file is to be found.” EX1006, 8:45-50. Cabrera discloses “hierarchical storage of data” (EX1005, 4:45-46) and a “master file table” (MFT) storing “all resident attributes of a file,” including the “system attributes ... used to store information needed or required by the system to perform its various functions.” *Id.*, 8:50-9:31. Cabrera does not expressly describe this MFT as a database, but a POSITA would have understood that attributes of a file stored in the MFT included file locations, as it was known that the location of a backup-archive copy needed to be tracked in order to facilitate transparent access to the backup-archive copy as it migrates (i.e., moves) through hierarchical storage locations. EX1006, 6:52-60. The POSITA reading Cabrera naturally would have looked to other backup-archive prior art, such as Whiting, for relevant teachings. The global directory index database of Whiting enables such transparent access by storing the location of backup data files. EX1006, 10:52-55, 21:47-49 (describing the record in the global database “which can be used to locate the actual file data contents”). The POSITA implementing Cabrera thus

would have naturally used these teachings from Whiting to provide implementation details for storing and updating the location of various archived versions of a file, including as the archive files are migrated from local storage to remote storage, so that the archive files can actually be located and accessed when needed by the system or user. EX1003, ¶156.

E. Claim Mapping

1. Claim 1

a) Element [1.1]⁴

If the preamble is limiting, Cabrera-Schneider-Whiting discloses it. Cabrera discloses “methods for transferring or archiving data from a local storage area to a remote storage area,” and discusses the archiving of such data in terms of files. EX1005, 1:7-9; *id.*, 9:43-10:21; *see also* EX1006, 1:12-14; EX1003, ¶157.

b) Element [1.2]

Cabrera-Schneider-Whiting renders this element obvious. Cabrera states that “any ... triggering event” may be used to trigger copying data from local storage to

⁴ The full claim language is included in mapping the claims for Ground 1. Hereinafter, Petitioner does not repeat the claim language but instead refers to it by the element/claim numbering used in the Listing of Claims (EX1023).

the staging area. *Id.*, 10:56-61; *see also id.*, 6:58-62 (discussing creation of “data that is to be backed up or archived”); EX1003, ¶158.

Schneider teaches, when “a request to write [to the disk] occurs, ... intercepting the write” to trigger copying the file on the disk before completing the write to the disk. EX1020, 12:5-8; *see also id.*, 30:4-6 (“How to Intercept Disk Activity”). Schneider further teaches that “applications” and “an operating system (OS) ... has the responsibility ... to save and recall information from a hard disk,” and that “applications” operating under the OS generate writes “during a write cycle (e.g., the saving of a document from a word processor or the saving of a photograph from an appropriate editor).” *Id.*, 1:20-22, 32:19-20. The operating system is a resident program, and so too the applications at least because they generate writes. *See* EX1001, 4:49-52 (defining “Resident Program”); EX1003, ¶159.

A POSITA would have been motivated to use detecting or intercepting “a request to write” to disk as taught by Schneider as a triggering event to initiate Cabrera’s backup-archival process, because doing so would enable the archiving system to “maintain a complete history of the changes made to ... a particular file.” EX1005, 1:32-34; *see also id.*, 2:5-9. Specifically, combining Cabrera and Schneider in this way increases the reliability of backup/archiving by “providing for a continuously running disk backup system” and “eliminat[ing] the need to pause and make backups or decide which files should be backed up in the context of short term

information recovery.” EX1020, 3:12-4, 5:24-25; *supra* Section IX.D; EX1003, ¶¶160-161.

c) Element [1.3]

Cabrera-Schneider-Whiting renders this element obvious. Cabrera discloses copying a file “from local storage to the staging file” in response to “any ... triggering event.” EX1005, 10:56-61. The staging file or staging area “is adapted for temporarily storing the data.” *Id.*, 7:3-7, 2:5-10 (“‘stage’ the information in a temporary storage area”). Cabrera discloses that “[a]ny mechanism that performs this function [of moving data] may be utilized such as, for example, reading the data from the appropriate location and then storing a copy of the data in the staging storage area[, ...] direct memory transfer and so forth.” *Id.*, 7:11-16. As discussed *supra* Sections IX.D and IX.E.1.b, it would have been obvious to initiate the backup-archive method of Cabrera responsive to interception of a request by an application or operating system to write to disk, as taught by Schneider, as the triggering event to initiate the backup-archive method of Cabrera. EX1003, ¶162.

Furthermore, a POSITA would have found it obvious to create the copy of the file temporally proximate to the operation being performed “so that a logically consistent backup copy is created” because creating such a copy before overwriting a file was a well-known, fundamental technique. EX1005, 1:51-55 (“[I]f a particular file or volume is to be backed up, it is usually important not to allow the contents of

the file or volume to change during the backup procedure so that a logically consistent backup copy is created.”); *see also* EX1004, 7:28-40 (describing creating a backup copy of data to be modified just before the data is modified); EX1020, 34:11-24, 35:3-8; EX1006, 1:23-29. EX1003, ¶¶163-164.

d) Element [1.4]

Cabrera-Schneider-Whiting renders this element obvious. Cabrera teaches monitoring the “amount of data in the staging area ... in order to identify when a backup or archive session should be initiated.” EX1005, 3:45-47. Cabrera further teaches an “event monitor 152” that “may be implemented in a wide variety of ways” in order “to watch for desired triggering events and to initiate appropriate action when the events occur.” EX1005, 13:1-10. Cabrera also teaches determining “whether more data should be appended to the staging file,” where “[t]his determination may be made based on any triggering event.” *Id.*, 11:22-27. A POSITA understood that evaluating whether to append more data to the staging file necessarily requires identifying the presence of the data in the staging file, and that this identifying step would occur responsive to the triggering event. EX1003, ¶165.

e) Element [1.5]

Cabrera teaches transmitting the archive file to a second storage location responsive to a second event. For example, Cabrera teaches, “[w]hen the second triggering event occurs and it is time to establish a connection to remote storage ...

the data should be transferred from the staging file to remote storage.” EX1005, 11:46-48; FIG. 5. The remote storage of Cabrera is a permanent storage, and thus a second storage location, because it is a persistent “backup or archive storage device.” EX1005, 6:61-62. The second event could include the “second triggering event” which “initiate[s] connection to remote storage” so that the file can be transferred (EX1005, 11:28-31), or an indication that the remote storage is connected (*id.*, 11:32-45). EX1003, ¶166.

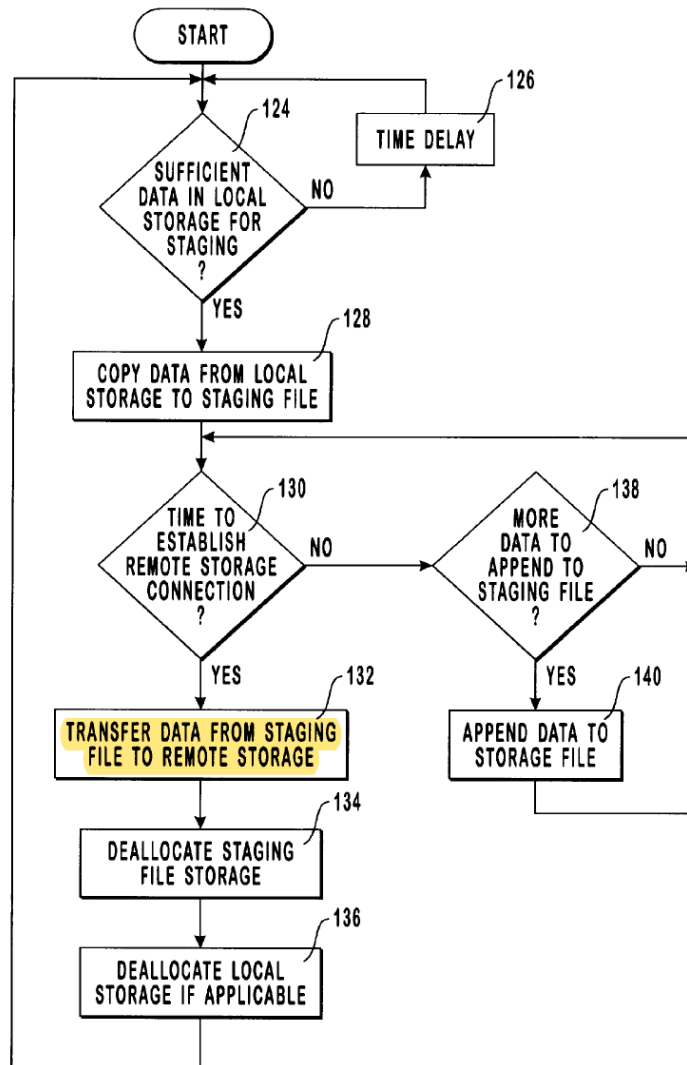


FIG. 5

EX1005, FIG. 5 (highlighting added).

f) Element [1.6]

Cabrera-Schneider-Whiting renders this element obvious. Cabrera discloses deallocating storage space in the local storage area after migrating the data to the staging area. EX1005, 14:9-15; FIG. 6. Whiting discloses a “backup directory file” that “indicate[s] the directory structure of the source disk volume, as well as pointers

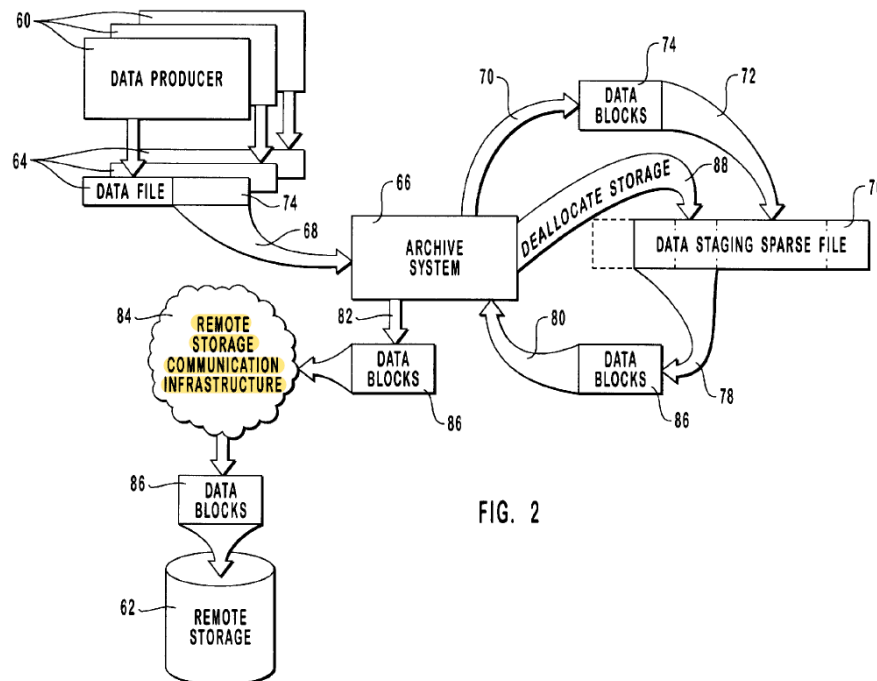
into the backup data files ... indicating where the data for each file is to be found.” EX1006, 8:45-50. As discussed *supra* Section IX.D, it would have been obvious to, after storing the backup data file in the staging area, update the backup directory file (database) taught by Whiting to indicate that the backup data file is located in the staging area, so that the data could be retrieved, for example, for transferring the data or using the data for recovery. EX1003, ¶167.

g) Element [1.7]

Cabrera-Schneider-Whiting renders this element obvious. Cabrera discloses transferring backup data “from the staging file to remote storage.” EX1005, 11:46-49. Cabrera further discloses that “a file stores data in a sequence of ... clusters” which are enumerated through “virtual cluster numbers (VCNs).” *Id.*, 9:44-54. Each VCN of a file “maps to a corresponding logical cluster number (LCN), which identifies the disk location of the cluster.” *Id.*, 9:55-62; *id.*, FIG. 4 (depicting VCNs and LCNs). A POSITA would have found it obvious to determine a final destination in the remote storage for the archive file so that the archive file could be written to a specific physical location within the remote storage. EX1005, 11:49-55 (explaining that transferring the file to remote storage includes “writing data to a ... disk”); EX1003, ¶169.

h) Element [1.8]

Cabrera-Schneider-Whiting renders this element obvious. Cabrera discloses that transferring data from the staging area to remote storage may be implemented by “delivering the data ... to an intermediate system or subsystem which will, in turn, deliver the appropriate data to remote storage.” EX1005, 7:52-60. This “intermediate system,” which is part of a “remote storage communication infrastructure” taught by Cabrera, is “an intermediate storage location” between the staging area and the remote storage. EX1005, 7:52-60; *id.*, 8:5-11, FIG. 2; EX1003, ¶170.



EX1005, FIG. 2 (highlighting added).

i) Elements [1.9]-[1.10]

As discussed above with regard to Element [1.6], Cabrera-Schneider-Whiting renders these elements obvious. *Supra* Section IX.E.1; EX1003, ¶171.

2. Claim 2

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.1 (claim 1). Cabrera discloses that “[i]n some embodiments ... remote storage 62 may be directly attached to the computer system where archive system 66 resides.” EX1005, 7:63-66, 6:49-52 (disclosing “backup or archive storage which is directly attached to a particular computer system”); EX1003, ¶172.

3. Claims 3-5

Cabrera-Schneider-Whiting render these claims obvious. *Supra* Section IX.E.1 (claim 1). Cabrera teaches “remote storage” for “backup or archive storage” that “may indeed be located at a remote location” such that “establishing a connection to the remote storage ... may involve establishing connections over networks, dial-up connections, connections through other computer systems, and so forth.” EX1005, 11:32-45, 11:49-55. Cabrera discloses a “networked environment” including a “memory storage device” of a remote computer device logically connected to a computer via a network. EX1005, 6:9-22, FIG. 1. Cabrera also discloses “a peer device” including a “memory storage device” as the remote storage. EX1005, 6:9-16. Cabrera further discloses a “[r]emote computer 49 [that]

may be another personal computer, a server, a router, a network PC, a peer device or other common network node” including “a memory storage device 50,” and connected to a computer in a networked environment such as “the Internet,” as the remote storage. EX1005, 6:9-22. EX1003, ¶173.

Cabrera thus teaches remote storage implemented as “a network attached storage device,” a “peer-to-peer storage device,” and “an internet storage area network” as defined by the '348 patent. *Compare* EX1005, 6:9-22, *with* EX1001, 4:34-40 (defining “Network Attached Storage Device” as “any storage device connected directly to a network to which a first computing device is also temporarily or permanently connected ...”), 4:45-48 (defining “Peer-to-Peer Storage Device” as “any storage area ... that can be accessed by the computing device when it is sharing resources with other network or Internet accessible computers”), 4:41-44 (defining “Internet storage area network” as “any storage area ... that can be accessed by the computing device when the computing device is temporarily or permanently connected to the Internet”); EX1003, ¶174.

4. Claim 6

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.1 (claim 1). Cabrera discloses implementing the backup-archival method in “the general context of computer-executable instructions, such as program modules, being executed by a personal computer.” EX1005, 5:4-6, 15:11-42 (claim 2)

(reciting a “computer program product ... comprising a computer readable medium for carrying executable instructions that are used to implement ... a method for transferring the data from the local storage area to the remote storage area”); EX1003, ¶175.

5. Claim 8

Elements [8.1]-[8.6] are identical to Elements [1.1]-[1.6], and are satisfied by Cabrera-Schneider-Whiting for the same reasons explained above for Elements [1.1]-[1.6]. *Supra* Section IX.E.1; EX1003, ¶176.

a) Elements [8.7]-[8.8]

Cabrera-Schneider-Whiting renders obvious moving the copied file (archive file) from the staging area (temporary storage location) to the remote storage communication infrastructure (intermediate storage location), and updating the global database to indicate the location of the copied file in the remote storage communication infrastructure. *Supra* Section IX.E.1. A POSITA would have understood that the remote storage communication infrastructure includes an output buffer to facilitate transferring data from the staging area to the remote storage. EX1003, ¶177.

b) Elements [8.9]-[8.10]

Cabrera-Schneider-Whiting renders obvious transmitting the copied file (archive file) from the remote storage communication infrastructure (output buffer)

to the remote storage (permanent storage location), and then updating the global directory database (database) to indicate that the copied file is in the remote storage.

Supra Section IX.E.1 ([1.5], [1.10]); EX1003, ¶178.

6. Claims 9-11

Claims 9-11 are identical to claims 2-3 and 5, respectively, and are obvious over Cabrera-Schneider-Whiting for the same reasons explained above for claims 2-3 and 5. *Supra* Sections IX.E.2, IX.E.3; EX1003, ¶179.

7. Claims 12-15

Elements [12.1]-[12.5], [13.1]-[13.5], [14.1]-[14.5], and [15.1]-[15.5] are identical to one another and to elements [1.1]-[1.5], and are obvious over Cabrera-Schneider-Whiting for the same reasons explained above for Elements [1.1]-[1.5]. *Supra* Section IX.E.1; EX1003, ¶180.

a) Element [12.6]

Cabrera discloses “establish[ing] a remote storage connection” in order to transfer the data to the remote storage. EX1005, 11:46-49. A POSITA would have understood that the transmission of the data to the remote storage taught by Cabrera is in response to a message indicating that the connection to the remote storage is established. EX1003, ¶181.

b) Element [13.6]

Cabrera discloses “the expiration of a fixed time” as a triggering event for establishing a connection to remote storage. EX1005, 11:10-20. EX1003, ¶182.

c) Element [14.6]

Cabrera discloses determining whether to append more data to the staging file “based on any triggering event,” such as the expiration of a timer. EX1005, 11:22-27. Determining whether to append more data to the staging file involves identifying the presence of the data in the staging file. *Supra* Section IX.E.1 ([1.5]). EX1003, ¶183.

d) Element [15.6]

As discussed with regard to Claim 1, the first and second events taught by Cabrera are separate events. *Supra* Section IX.E.1 ([1.4-1.5]). EX1003, ¶184.

8. Claim 16

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). Cabrera discloses “an operating system 35, one or more application programs 36, [and] other program modules 37,” and a POSITA would have

understood that the “data producer” of Cabrera would include any of these elements. EX1005, 5:58-62, 6:58-66; EX1003, ¶185.

9. Claim 17

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.8 (claim 16). A POSITA would have understood that the data producer of Cabrera comprises a program having control over file operations on one or more operating files. EX1005, 5:58-62; *id.*, 6:58-66; *supra* Section IX.E.1 ([1.2-3]); EX1003, ¶186.

10. Claim 18

Cabrera-Schneider-Whiting renders this claim obvious because the file-manipulation operation taught by Schneider is a “write” operation. EX1020, 12:5-8; *supra* Section IX.E.7 (claim 15); EX1003, ¶187.

11. Claim 19

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15); EX1005, 6:66-67 (disclosing a “[d]ata file”). Cabrera discloses archiving a log file, which is a system file. EX1005, 12:37-40 (discussing archiving a log file); EX1008, 38 (explaining that “[t]he log file is a system file”). Cabrera further discloses files with both system attributes and user attributes. EX1005, 8:50-9:3; EX1003, ¶188.

12. Claim 20

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). A POSITA would have found it obvious that the archive file comprises at least some of the operating file in order to backup/archive the file for later recovery. *Supra* Section IX.E.1 ([1.2-3]); EX1003, ¶189.

13. Claim 23

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). Cabrera discloses that “[a]ny type of technology may be utilized to implement” the staging area. EX1005, 7:7-11, 7:40-41. Cabrera discloses a variety of storage devices internal to the computing device. *Id.*, 5:21-57, FIG. 1. A POSITA would have found it obvious to implement the staging area (temporary storage location) in a storage device internal to the computing device because Cabrera discloses staging data “prior to transfer to remote storage.” *Id.*, 13:32-34; EX1003, ¶190.

14. Claim 24

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.13 (claim 23). A POSITA would have understood that a buffer is appropriate

for staging the data prior to transfer because “a buffer is a temporary storage location for information being sent or received.” EX1009, 132; EX1003, ¶191.

15. Claim 25

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). Cabrera discloses several storage devices that “store data ... accessible by a computer” (EX1005, 5:50-57) and teaches implementing the staging area, which is for temporarily storing the data,” with “[a]ny type of technology” (*id.*, 7:7-11, 7:40-41). “[T]he data may be copied or moved *from* local storage *to* a staging area,” indicating that the staging area may be separate from the computer’s local memory. *Id.*, 3:41-42 (emphasis added). A POSITA therefore would have naturally implemented the staging area (temporary storage) in an external storage device to free up local storage space on the computer device. *Id.*, 3:61-67 (describing deallocating “storage space from the local storage area”). There was a reasonable expectation of success in doing so at least because Cabrera expressly discloses storage areas for the backup-archive process that are “separate and apart from the local storage volumes.” *Id.*, 2:66-3:1; EX1003, ¶192.

16. Claim 26

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). Cabrera discloses identifying the presence of data in the staging file responsive to “any triggering event.” EX1005, 11:22-27; *supra* Section IX.E.1

([1.4]). Cabrera discloses triggering events that include “the expiration of a given time delay.” EX1005, 10:56-60. A POSITA therefore would have found it obvious, with a reasonable expectation of success, to implement Cabrera where the first event comprises a message from a timer because Cabrera provides a finite list of triggering events and indicates that any one of them would be suitable. EX1003, ¶193.

17. Claim 27

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). It teaches “a second triggering event” for transferring the archive file from the staging area to remote storage, and explains that “this triggering event may be one of several things” such as “a command received from an outside source” or “the expiration of a fixed time.” EX1005, 11:10-16. A POSITA would have understood that “the expiration of a fixed time” involves a “a message from a timer.” EX1003, ¶194.

18. Claim 28

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). Cabrera discloses that the permanent storage location is located in one of a personal attached storage device, a network attached storage device, an Internet storage area network, or a peer-to-peer storage device. *Supra* Sections IX.E.2-3; EX1003, ¶195.

19. Claim 29

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). Cabrera also discloses “[e]mbodiments ... includ[ing] computer readable media having executable instructions ... stored thereon.” EX1005, 4:50-67. Whiting discloses implementing the method as “backup software installed” on a computer. EX1006, 35:8. EX1003, ¶196.

20. Claim 30

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). Cabrera discloses a “networked environment” including a “memory storage device” of a remote computer device logically connected to a computer via a network, such as the Internet. EX1005, 6:9-22, FIG. 1; EX1003, ¶197.

21. Claim 31

Cabrera-Schneider-Whiting renders this claim obvious. *Supra* Section IX.E.7 (claim 15). The second storage location of Cabrera is the permanent storage location. *Supra* Section IX.E.1 ([1.5]); EX1003, ¶198.

X. NO OBJECTIVE INDICIA OF NON-OBVIOUSNESS

There are no objective indicia of non-obviousness as to any challenged claim.

XI. CONCLUSION

Petitioners request institution for claims 1-6, 8-20, and 23-31, on Grounds 1 and 2 specified in this petition.

Respectfully submitted,

October 31, 2023

By: /Andrew M. Mason/

Andrew M. Mason (Reg. No. 64,034)
andrew.mason@klarquist.com
KLARQUIST SPARKMAN, LLP
One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Tel: 503-595-5300
Fax: 503-595-5301

Counsel for Petitioners

**CERTIFICATE OF COMPLIANCE WITH
TYPE-VOLUME LIMITATION PURSUANT TO 37 C.F.R. § 42.24**

This brief complies with the type-volume limitation of 37 C.F.R. § 42.24(a)(1)(i).

The brief contains 13,962 words, excluding the parts of the brief exempted by 37 C.F.R. § 42.24(a).

The brief has been prepared in a proportionally spaced typeface using Microsoft Word 2013 in a 14-point Times New Roman font.

October 31, 2023

By: /Andrew M. Mason/
Andrew M. Mason (Reg. No. 64,034)
andrew.mason@klarquist.com
KLARQUIST SPARKMAN, LLP
One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Tel: 503-595-5300
Fax: 503-595-5301

Counsel for Petitioners

CERTIFICATE OF SERVICE
IN COMPLIANCE WITH 37 C.F.R. § 42.6(e)(4)

The undersigned certifies that on October 31, 2023, the **Petition for *Inter Partes* Review of U.S. Patent No. 9,218,348 and Exhibits 1001 – 1023** were served via **Express Mail** on the Patent Owner at the following address of record as listed on the USPTO Patent Center:

DATANET, LLC
c/o Messner Reeves LLP
7250 N. 16th Street, Suite 410
Phoenix, AZ 85020

By: /Andrew M. Mason/
Andrew M. Mason (Reg. No. 64,034)
andrew.mason@klarquist.com
KLARQUIST SPARKMAN, LLP
One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Tel: 503-595-5300
Fax: 503-595-5301

Counsel for Petitioners