

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

DROPBOX, INC. AND MICROSOFT CORPORATION,

Petitioners,

v.

DATANET, LLC,

Patent Owner.

Case IPR2024-00078

U.S. Patent Nos. 8,473,478

**DECLARATION OF ZAYDOON (“JAY”) JAWADI
IN SUPPORT OF PATENT OWNER’S RESPONSE**

IPR2024-00078

Exhibit 2020

Dropbox, Inc. and Microsoft Corporation v. Datamet, LLC

TABLE OF CONTENTS

I. Introduction.....	1
A. Engagement and Compensation	1
B. Background and Qualifications	1
C. Case Materials Reviewed	7
D. Extrinsic Materials Considered	7
E. Legal Understanding	9
a. Claim Construction.....	9
b. Anticipation	9
c. Obviousness.....	10
d. Person of Ordinary Skill in the Art	14
II. Overview of The '478 PATENT	15
A. Data Backup and Management.....	17
B. Disk Drive Sectors versus Files.....	20
a. Computer Data Storage Layers	20
b. Disk Drive Sectors.....	21
c. Files	23
d. BIOS	26
III. Claim Construction	29
a. “Searching”	29
IV. Ground 1 - Long and Kenley	31
A. Long Overview	31
a. Long is based on disk drive sectors, not files.....	31
b. Long handles sectors containing file data or non-file data but does so on sector basis.....	34
i. Long discloses disk drive tracks.....	34
ii. Long describes dealing with data storage devices, but not files and not file systems.	35

iii.	Long’s operations and events relate to data storage devices, not files.	37
iv.	Long is implemented as part of the BIOS or as a memory resident program.....	39
v.	Long teaches detecting access requests to alter data on storage device locations (sectors) but not files.	40
c.	A POSITA would have understood Long as teaching away from file system implementation.	41
d.	In summary, Long is sector based, not file based, and not applicable to files as contemplated in Petitioners’ Expert’s Declaration.	42
B.	Kenley Overview.....	43
C.	A POSITA would not have combined Kenley and Long.....	43
a.	Long’s teachings specific to sector-level implementation could not be reasonably expected to succeed in file-level implementation and would have contradicted Long’s basic teachings.....	43
b.	Long is implemented as part of BIOS or resident program, both sector-based, low-level.	45
c.	The proposed Kenley-Long combination would have required substantial redesign and reconstruction of Long.	45
d.	A POSITA would have recognized that the proposed modifications would have changed the principle of operation of the modified reference...46	
e.	Petitioners’ Expert’s Declaration at ¶ 25 states that “I understand that in considering obviousness, it is important not to determine obviousness using the benefit of hindsight derived from the descriptions found in the patent being considered.”	47
D.	The Petition and Petitioners’ Expert Declaration fail to show that Limitation 1.4 is met by the proposed Kenley-Long combination.....	48
a.	Long’s flushing of the memory buffer does not disclose a search for any alleged archive file.....	48
b.	Long’s semaphoring the memory buffer does not indicate a search for the alleged archive files.	54
c.	The proposed combination with Kenley does not cure the above deficiencies of Long.	55

E.	The Petition and Petitioners’ Expert Declaration fail to show that Limitation 1.6 is met by the proposed Kenley-Long combination.....	55
F.	The Petition and Petitioners’ Expert Declaration fail to show that Limitation 1.9 is met by the proposed Kenley-Long combination.....	61
G.	The Petition and Petitioners’ Expert Declaration fail to show that Limitation 1.10 is met by the proposed Kenley-Long combination.....	63
V.	Ground 2 - Cabrera, Schneider, and Whiting	65
A.	The Petition fails to show that Limitation 1.4 is disclosed or suggested by Ground 2’s proposed combination.....	65
a.	The cited Cabrera disclosures do not disclose searching for the archive file responsive to a first event.....	65
b.	The staging file of Cabrera does not correspond to the claimed archive file	67
c.	The “staging file” is only disclosed in conjunction with Cabrera’s Figure 5, and does not differ from the “staging area” – a conglomeration of data from the local disk.	69
B.	The Petition fails to show that Limitation 1.5 is disclosed or suggested by Cabrera-Schneider-Whiting	72
a.	Ground 2 uses Cabrera’s intermediate system to meet Limitation 1.8... ..	72
b.	Limitation 1.5 requires moving the archive file to a second / permanent storage location responsive to a second event.....	73
c.	Cabrera is silent on the when or what might trigger moving data from the intermediate system to remote storage.	74
C.	A POSITA would not have combined Cabrera and Schneider as in Petitioners’ Expert Declaration because doing so would not work and would fundamentally alter the operation of Cabrera.	76
a.	Cabrera is periodic in nature.	76
b.	The proposed combination would not work with the system of Cabrera.....	77
D.	Whiting does not disclose or suggest updating a database to reflect storage of an archive file in multiple locations and a POSITA would not combine Whiting with Schneider and Cabrera.....	81

VII. Conclusion 84

I, Zaydoon (“Jay”) Jawadi, declare as follows:

I. Introduction

1. My name is Zaydoon (“Jay”) Jawadi.
2. I am an independent expert and consultant.

A. Engagement and Compensation

3. I have been retained as an expert witness on behalf of Datanet, LLC (“Datanet”) for the above-captioned *Inter Partes* Review (IPR) case regarding U.S. Patent No. 8,473,478 (“’478 Patent”).

4. I am being compensated for my work and any travel expenses in connection with this proceeding at my standard consulting rates. My compensation is in no way dependent on or contingent on the outcome of my analysis or opinions rendered in this proceeding and is in no way dependent on or contingent on the results of these or any other proceedings relating to the above-captioned patent.

B. Background and Qualifications

5. As shown in my curriculum vitae (attached as Exhibit 2021), I have a Master of Science in Computer Science from Columbia University with a Citation for Outstanding Achievement – Dean’s Honor Student, a Bachelor of Science in Electrical Engineering from Mosul University, and over 40 years of experience in software and product design and development, engineering, consulting, and

management in the fields of data storage, Internet, software, data networking, computing systems, and telecommunication.

6. I have worked with and possess expertise in numerous technologies, including system and embedded software, data storage technologies and interfaces, Internet and website technologies, databases, data networking technologies and protocols, and telephony.

7. From 1978 to 1980, I worked as a telecommunication/electrical engineer for Emirtel (formerly Cable and Wireless, now Etisalat). During my employment at Emirtel, among other things, I worked on telephony and telecommunication products and services, and I designed and developed software in assembly and high-level languages for archiving, storing, and retrieving data to and from data storage devices, such as disk drives and tape drives. I also maintained and operated software programs for critical data archival, backup, storage, and retrieval using data storage devices, such as disk drives and tape drives.

8. From 1981 to 1983, I worked as a software engineer for Amdahl Corporation (now Fujitsu), a California-based Kenley supplier of computers, systems, and data storage subsystems. Among my responsibilities, I designed and developed bug tracking software using relational database.

9. From 1984 to 1994, I worked as a software, data storage, and systems consultant to various data storage and computer companies in California, the United

States, Asia, and Europe. I provided technical consulting services in database design and development, data storage, data storage systems, data storage devices, software design and development, system software, device driver software, data storage device firmware, disk drive controller firmware, data storage software, data storage chips, data storage tools, data storage test systems and test software, SCSI, IDE, data storage and I/O protocol development systems, data storage controllers, data storage and I/O protocol analyzers, data storage and I/O monitoring systems, and data storage manufacturing systems and software. In one of my assignments, I was the lead developer and team leader of a commercial relational database development project for one of the top 25 companies (then and now) of America's Fortune-500 list.

10. From 1992 to 1996, I was President and founder of Zadian Technologies, Inc., a California-based leading supplier of networked data storage test systems, with over 50,000 units installed worldwide in mission-critical customer operations with premier high-technology customers, such as Conner Peripherals (now Seagate), DEC (now HP), EMC (now Dell EMC), Exabyte, Fujitsu, HP, IBM, Intel, Iomega, Quantum (now Seagate), Seagate, Sony, StorageTek, SyQuest, Tandberg, Tandem (now HP), Toshiba, Unisys, and WD. The company's products included test systems, manufacturing systems, and development systems for data storage devices (disk drives, tape drives, removable drives, flash drives, optical

drives, CD-ROM drives, Jukeboxes, and RAID) and data storage interfaces (SCSI, ATA / IDE / ATAPI, Fibre Channel, SSA, and PCMCIA / PC Card).

11. In 1996, Zadian Technologies was acquired by UK-based Xyratex International LTD (NASDAQ: XRTX, which was later acquired by Seagate, NASDAQ: STX, in 2014). Following Zadian's acquisition by Xyratex, I became an employee of Xyratex until 1998. At Xyratex, I was a general manager of a data storage interface business unit and, subsequently, a general manager of a data networking analysis tools business unit, which designed and built Gigabit Ethernet network protocol analysis and monitoring products, which were sold, under OEM agreement, by the largest supplier of network protocol analysis and monitoring products.

12. From 1999 to 2001, I was CEO, Chairman, and cofounder of Can Do, Inc., a California-based Internet eCommerce and community company. The CanDo.com website offered over 10,000 products for sale as well as extensive consumer features, such as news, chat, messages, and product information for people with disabilities. The company also provided technologies for display magnification and sound/audio adaptation through the Internet to make websites more accessible to persons with vision and hearing impairments. The company was funded by leading venture capital firms.

13. From 2001 to 2007, I was President and cofounder of CoAssure, Inc., a California-based provider of Web-based technology services and solutions for automated telephony speech recognition and touchtone applications, serving multiple Fortune-500 companies. The system was delivered as Software as a Service (SaaS). Among other things, I was responsible for system and file archival, backup, restoration, and management.

14. From 2009 to 2022, I was President and cofounder of Rate Speeches, Inc., a California-based Internet company providing online services, resources, and technologies for creating, rating, evaluating, and enhancing presentation, and communication skills. Rate Speeches also operated the ratespeeches.com website and the Speech Evaluator online software. The system included functionality delivered as Software as a Service (SaaS). In my career, I have designed, developed, tested, and deployed as well as overseen the design, development, testing, and deployment of multiple complex software, software applications, system software, websites, and systems.

15. Since moving to Silicon Valley in Northern California in 1981, I have worked on numerous technology products that have generated billions of dollars in sales for my employers and customers.

16. I hold a California community college lifelong computer science instructor credential. I have taught various data storage and computer technologies,

including SCSI, IDE (EIDE/ATA), SSA, and Fibre Channel, to thousands of professional engineers and academic students in the United States, Europe, and Asia.

17. Additionally, in my work as an expert and consultant, I have examined, analyzed, and inspected numerous data storage systems, computer systems, software products, database software, and websites, spanning tens of millions of lines of software source code.

18. By September 2000, I had more than two decades of expertise directly applicable to the technology at issue in this case, including expertise in file management; archival, backup, and restoration; BIOS programming; device drivers; interrupt handlers; interrupt 13h (INT 13h) handlers; trapping / intercepting / detecting device drivers and BIOS interrupts; resident programs and terminate-and-stay-resident programs; disk drive firmware; SCSI, IDE, SSA, and Fibre Channel; disk drives, tape drives, optical drives, and other storage devices; disk drive controllers and host adapters; database and database application design and development; as well as CP/M, DOS, Windows, Unix, and other operating systems.

19. My additional industry experience is in my curriculum vitae.

20. My expert litigation support cases, including cases in which I have testified during the last four years as an expert, can be found in my curriculum vitae, which is Exhibit 2021.

21. In light of my education, training, knowledge, and experience, I am qualified to provide opinions regarding the state of the art at the time of the invention (or the priority date) of the '478 Patent, which I understand to be no later than September 21, 2000, and I am qualified to provide opinions regarding how a person of ordinary skill in the art ("POSITA") at that time would have interpreted and understood the '478 Patent.

C. Case Materials Reviewed

22. In preparing this Declaration, I reviewed the '478 Patent, including its claims in view of its specification, the prosecution history of the '478 Patent, and the IPR2024-00078 Petition ("Petition") and its exhibits, including Petitioners' Expert's Declaration ("EX1003" or the "Petitioners' Expert Declaration") and the Transcript of the Deposition of Petitioners' Expert EX2022, and the Institution Decision (Paper 9).

D. Extrinsic Materials Considered

23. In preparing this Declaration, I also reviewed the following extrinsic technical references, from the time of the invention, attached as exhibits:

Exhibit	Description
EX2003	Computer Busses, Bill Buchanan, CRC Press, 2000, pp. 139-143
EX2004	The Multi-boot Configuration Handbook, Roderick W. Smith, Que, 2000, pp. 71, 76-79

Exhibit	Description
EX2005	Encyclopedia of Microcomputers, Volume 2 - Authoring Systems for Interactive Video to Compiler Design; Allen Kent, James G. Williams, and Rosalind Kent; Taylor & Francis; 1988; pp. 196-197
EX2006	PhoenixBIOS 4.0 Revision 6 User's Manual; Phoenix Technologies Ltd.; June 22, 2000; pp. ii-iv, 31-34, 39, 43-51, 80
EX2007	Forensic Computing: A Practitioner's Guide; A. J. Sammes, Brian Jenkinson; Springer Science & Business Media; 2000; pp. 119; 132, 145
EX2008	Introduction to Software Engineering; Ronald J. Leach; CRC-Press; 2000; p. 292
EX2009	PC Interrupts: A Programmer's Reference to BIOS, DOS, and Thirdparty Calls; Ralf Brown, James Kyle; Addison-Wesley; 1991
EX2010	Hard Disk Controller: the Disk Drive's Brain and Body; James Jeppesen, Walt Allen, Steve Anderson, Michael Pils!; Infineon Technologies; 2001
EX2011	SCSI-2 Standard; ANSI; 1994
EX2012	ATA Draft Standard; ANSI
EX2013	Newton's Telecom Dictionary: The Official Dictionary of Telecommunications Networking and Internet; 16th Edition; Harry Newton, Ray Horak; Cmp Books; 2000
EX2014	U.S. Patent Publication No. 2009/0254594 to Laurion D. Burchall
EX2016	U.S. Patent Publication No. 2007/0226436 to Cheng et al.
EX2017	U.S. Patent Publication No. 2004/0003103 to Witt et al.
EX2018	Foundations of Computer Technology; A. John Anderson; 1994; Taylor & Francis
EX2022	Deposition of David Maier, Ph.D., July 18, 2024
EX2023	Microsoft Press, Computer Dictionary, Third Edition (1997), p. 424
EX2024	The International Dictionary of Data Communications; Robert A. Saigh; 1998; Glenlake Publishing Company; pp. 159, 234, 278

E. Legal Understanding

24. I am not a lawyer. In forming my opinions as set forth in this Declaration, I have relied upon legal principles supplied to me by attorneys for Datanet.

a. Claim Construction

25. I understand that determining whether a patent claim is valid requires a two-step analysis. First, the claim must be construed. Second, the properly construed claim must be compared to the prior art.

26. With respect to the first step, I understand that claims are construed from the perspective of a POSITA at the time of the purported invention. I understand that in the context of an inter partes review proceeding, absent a disclaimer or express definition, claim terms are given their plain and ordinary meaning to a POSITA in view of the specification and file history and the knowledge of a POSITA. Unless otherwise noted, I have applied what I consider to be the plain and ordinary meaning of all terms as understood by a POSITA at the time of the priority date of each patent discussed above.

b. Anticipation

27. I understand that a patent claim is presumed to be valid. I understand that to overcome that presumption in an inter partes review proceeding, the challenger must show that the claim is invalid by a preponderance of the evidence. I

understand the preponderance of the evidence to require the greater weight of evidence in favor of the challenger.

28. I understand that, although not expressly disclosed, subject matter may be inherently disclosed in a prior art reference where that subject matter is necessarily present in the subject matter disclosed and would be understood to be so by those of ordinary skill in the art. I understand that the fact that a certain result or characteristic may occur or be present in the prior art is not sufficient to establish the inherency of that result or characteristic.

29. I understand that a claim is invalid if each and every element of the claim is disclosed in a single prior art reference with the elements arranged as specified in the claim, either expressly or inherently. I understand that invalidity based on the disclosure of a single prior art reference in this manner is called anticipation.

c. Obviousness

30. I understand that a claim would have been obvious under 35 U.S.C. § 103 if one or more prior art references disclose, expressly or inherently, every claim limitation in such a way so as to render the claim, as a whole, obvious to a person of ordinary skill in the art at the time the purported invention was made. The relevant standard for obviousness is as follows:

A patent may not be obtained though the invention is not

identically disclosed or described as set forth in section 102, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

35 U.S.C. § 103 (pre-AIA).

31. In determining whether or not a patented invention would have been obvious, the following factors should be considered: (a) the scope and content of the prior art; (b) the differences between the prior art and the claims at issue; (c) the level of ordinary skill in the art; and (d) whatever “secondary considerations” may be present.

32. I understand that certain “secondary considerations” may be relevant in determining whether or not an invention would have been obvious, and that these secondary considerations may include commercial success of a product using the invention, if that commercial success is due to the invention; long-felt but unsolved need for the invention; evidence of copying of the claimed invention by others; industry acceptance; initial skepticism of the invention; failure of others to solve the problem; praise of the invention by others; the taking of licenses under the patent by others; and simultaneous invention by others (which may be an indication that the claimed invention was the result of ordinary skill in the art).

33. I understand that a patent composed of several elements is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art. While multiple prior art references or elements may, in some circumstances, be combined to render a patent claim obvious, I understand that I should consider whether there is an apparent reason to combine the prior art references or elements in the way the patent claims. To determine whether such an apparent reason exists to combine the prior art references or elements in the way a patent claims, it will often be necessary to look to the interrelated teaching of multiple patents, to the effects of demands known to the design community or present in the marketplace, and to the background knowledge possessed by a person having ordinary skill in the art.

34. I also understand that when the prior art teaches away from combining prior art references or certain known elements, discovery of a successful means of combining them is more likely to be non-obvious. A prior art reference may be said to teach away from a patent when a person of ordinary skill, upon reading the reference, would be discouraged from following the path set out in the patent or would be led in a direction divergent from the path that was taken by the patent. Evidence showing there was no reasonable expectation of succeeding in combining references, that the proposed modification or combination of the prior art would change the principle of operation of the prior art invention, or that the suggested

combination of references would require a substantial reconstruction and redesign of the elements may support a conclusion of non-obviousness. I understand that to be applicable a prior art reference must be analogous art to the claimed invention. A reference is analogous art to the claimed invention if: (1) the reference is from the same field of endeavor as the claimed invention (even if it addresses a different problem); or (2) the reference is reasonably pertinent to the problem faced by the inventor (even if it is not in the same field of endeavor as the claimed invention).

35. I also understand that it is not permissible to use hindsight in assessing whether a claimed invention is obvious. Rather, I understand that, to assess obviousness, one must place oneself in the shoes of a person having ordinary skill in the relevant field of technology at the time the inventions were made who is trying to address the issues or solve the problems faced by the inventor and ignore the knowledge one currently now has of the inventions.

36. I also understand that there are numerous ways in which to articulate the legal standard for obviousness, including: (1) combining prior art elements according to known methods to yield predictable results, (2) simple substitution of one known element for another to obtain predictable results, (3) use of a known technique to improve similar devices (methods, or products) in the same way, (4) applying a known technique to a known device, method, or product ready for improvement to yield predictable results, (5) choosing from a finite number of

identified, predictable solutions with a reasonable expectation of success, (6) known work in one field of endeavor may prompt variations of the work for use in either the same field or a different one based on design incentives or other market forces if the variations are predictable to one of ordinary skill in the art, and (7) some teaching, suggestion, or motivation in the prior art that would have led one of ordinary skill to modify the prior art reference or to combine prior art reference teachings to arrive at the claimed invention.

d. Person of Ordinary Skill in the Art

37. I understand that the meaning of claim terms and a patent's validity are to be determined from the vantage point of one of ordinary skill in the art at the time of the invention. In determining who would be one of such ordinary skill, I understand it is appropriate to consider criteria such as: (a) the type of problems encountered in the art; (b) prior art solutions to those problems; (c) the rapidity with which innovations are made; (d) the sophistication of the technology; and (e) the education level of active workers in the field.

38. Although I am not rendering an opinion about the level of skill of a POSITA proffered in Petitioners' Expert's Declaration, based on my professional experience, I have an understanding of the capabilities of a POSITA (as such a POSITA is defined in Petitioners' Expert's Declarations in paragraph 31 of each declaration).

39. For purposes of this declaration only, I adopt Petitioners' the definition of a POSITA as set forth in EX1003 ¶ 31.

40. Over the course of my career, I have supervised and directed many such persons. Additionally, I myself was qualified as at least a POSITA, under Petitioners' definition, at the time that the '478 Patent was filed and at its priority date.

II. Overview of The '478 PATENT

41. I provide below a brief overview of features of the '478 Patent that are relevant to the arguments below.

42. The '478 Patent deals with file-system level files (rather than disk-drive level sectors) and operates at the file level (not sector level).

43. Each of the independent claims of the '478 Patent recites the word file, specifically the term "operating file" and the term "archive file."

detecting an instruction by an operating system to perform an operation on an **operating file**;

creating an archive file from the operating file and storing the **archive file**

'478 Patent, Claims 1, 8, 9, 10, 11 (emphasis added);

44. The Abstract of the '478 Patent describes "[a] method for **archiving files** includes determining when a change in an **operating file** is imminent, capturing the **operating file** immediately before the change in the **operating file** occurs, if the

operating file has not already been captured; and capturing the **operating file** immediately after the change in the **operating file** has occurred.” (Emphasis added.)

45. The word file(s) appears 196 times in the specification of each of the ’478 Patent and appears 47 times in the claims of the ’478 Patent.

46. The ’478 Patent describes operations specific to files (not sectors), such as “open,” “close,” “rename,” and “delete.” ’478 Patent, 4:46-47, 6:54, 7:28, 7:57.

47. The words sector(s), track (other than “keep track”), head, and block (referring to storage unit) do not appear in the ’478 Patent.

48. The ’478 Patent provides near real-time backup operations even when backup storage devices are temporarily unavailable. EX1001, 6:34-43. The ’478 Patent discloses actively monitoring a computer’s operating system for, e.g., requests to save a modified file. EX1001, 7:11-7:27. When those operating system activities are detected, the technology initiates, in near real time, a backup operation to a local queue or buffer. *Id.* When a suitable backup storage device becomes available, the modified file contents are transmitted to the backup storage device for long-term storage. EX1001, 7:12-7:27.

49. I provide a brief overview of some relevant technical background below.

A. Data Backup and Management

50. In my data storage experience in the 1980s and 1990s, I witnessed the profound need for data backup and archival solutions.

51. As explained in the '478 Patent (1:16-2:55), data protection and management have been a challenge for decades. In the 1990s, several factors heightened the vulnerability to data loss and, consequently, intensified the need for backup and archival solutions. These factors included the broadened popularity of home and office computers; the proliferation of computing devices, such as notebook computers and PDAs; the rapid and continual increase in data storage capacity; and the critical dependence on servers spearheaded by the wide adoption of networking and the Internet.

52. Data loss can result from hardware failures, software failures, fires, stolen equipment, viruses, and user errors (accidental file deletions, file overwrites, errant programs, etc.).

53. Backup and archival solutions relied on data storage devices and subsystems, such as removable disk drives, flash drives, tape drives, optical drives, jukeboxes (AKA media changers or libraries; backup/archive devices that automatically load/unload removable storage media, such as tapes or optical disks), and disk arrays (RAID).

54. The backup device(s) may be local to the computer (such as a second disk drive or a flash drive embedded in the computer), external and on a local area network shared through a server, or remote and shared through wide area network.

55. Most commonly, the backup / archival was scheduled, manual, or based on mirroring or similar technology.

56. Two major classes of data backup and archival existed around the year 2000 and still exist today.

57. The first class of archival solution is sector (or block) based, where data of disk drive sectors is backed up on the same disk drive, on another disk drive, and/or on another type of storage device, such as tape drive or Jukebox.

58. With sector-based backup / archive systems, a major advantage is the ability to restore the entire disk drive if needed, for example if the local disk drive fails or is accidentally erased.

59. A major disadvantage of sector-based backup / archive is its unsuitability for recovering only a subset of the disk drive data, particularly files. Archiving and retrieving files requires that the archive software know how files are stored on the disk drive and how to access them. However, sector-based archival solutions operate at the sector level and rely on BIOS or low-level device drivers, which function at a lower level than the operating system and file system. In other

words, there is no way to access files through BIOS or low-level device drivers, or even to distinguish file from non-file data on individual sectors.

60. The other class of archival solution is file based, where files are backed up on the same disk drive, on another disk drive, and/or on another type of storage device, such as tape drive or Jukebox.

61. With file-based backup / archive systems, a major advantage is the ability to restore individual selected files if needed, for example if a file is accidentally corrupted or deleted.

62. A major disadvantage of file-based backup / archive is the inability to restore non-file data or the entire disk drive if needed, for example if the local disk drive fails or is accidentally erased. Archiving and retrieving sectors requires that the archive software must read sectors. However, file-based archival solutions operate at the operating system and file system level and rely on the high-level operating system services. Non file-system data (such as the master boot record and the boot sector) generally cannot be archived or restored with file-based solutions. EX2007, p. 145.

63. In summary, sector-based archive solutions are different and incompatible with file-based archive solutions. Both solutions still exist separately today because of the difficulty in providing any sort of combined solution which can address the shortcomings inherent in each.

B. Disk Drive Sectors versus Files

a. Computer Data Storage Layers

64. Computer data storage follows a layered architecture, with each layer hiding the details of the next lower layer. EX1015, pp. 390-391.

65. At the highest level are application programs, such as Microsoft Word, which allow the user to create, edit, and process files. Application programs call the operating system for system services, such as creation of files and accessing data storage services. The operating system (specifically the file system) manages the directories (folders) and files, including allocating space and providing access to directories and files. To access data storage devices, such as a disk drive, the operating system calls low-level device drivers (or BIOS), which communicate with the data storage devices through interfaces, such as SCSI or IDE, using sector (or block) level commands, not files. In other words, file systems and disk sectors exist at separate conceptual levels in computer data storage architectures. File systems operate at the operating system level, are hardware agnostic (implementing the same functionality regardless of the type of data storage device used), and call hardware-specific disk drive device drivers to access the data stored in sectors (blocks). EX2005, p. 197; EX2018, pp. 277, 282, 288-290.

66. For at least the above reasons, the techniques and solutions for managing data at the file system level are rarely applicable to the disk/sector level, and vice versa.

67. A POSITA would have known that computer data storage follows a layered architecture, with each layer hiding the details of the next lower layer.

b. Disk Drive Sectors

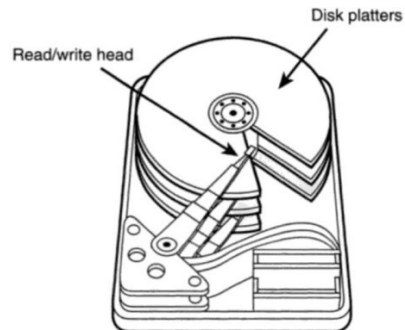
68. In this Declaration, I refer to sectors as pertaining to disk drives.

69. Disk drives are also called hard disk drives or hard drives to distinguish them from flexible disk drives or floppy drives.

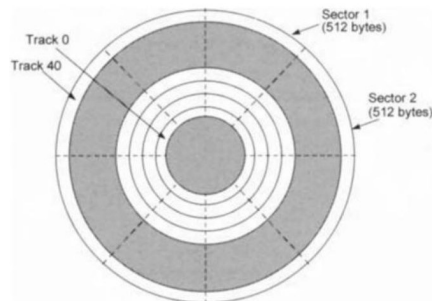
70. A disk drive consists of media platters coated with magnetic material.

71. The surface of each platter is formatted to contain a series of concentric circles called tracks. Each track is segmented into and formatted to contain smaller data storage units called sectors. Each sector typically contains 512 bytes of data. For each platter surface, a dedicated head writes and reads data to and from sectors. The heads of a disk drive are constructed and affixed together as a head assembly, moving together, seeking to tracks, and writing and reading data to and from sectors. When the heads are at a particular location, the collection of tracks in that location is called a cylinder; in other words, a cylinder is a collection of tracks in a particular head position; the cylinder is given its name, because the track configuration or

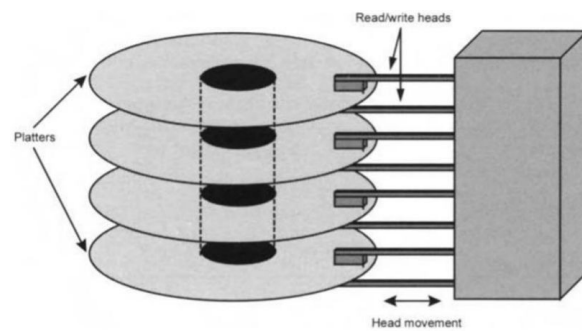
collection for any track location resembles a cylinder. EX2003, pp. 139-141;
EX2004, p. 76.



EX2004, p. 76



EX2003, pp. 140



EX2003, pp. 141

72. The term block (AKA logical block) refers to a unit of data transfer, typically one or multiple sectors.

73. Host computers exchange data with disk drives through data storage interfaces. In 2000, the two most common data storage interfaces were ATA (AKA IDE or EIDE) (AT Attachment; Integrated Drive Electronics; Enhanced IDE) and SCSI (Small Computer System Interface). EX2004, pp. 76-79.

74. Depending on the disk drive interface, read and write commands use sectors or blocks as units of data transfer. To read or write a disk drive sector, the read or write command specifies the sector location as cylinder-head-sector (CHS).

Alternatively, to read or write a disk drive block (logical block), the read or write command specifies the block location as logical block address (LBA). ATA used CHS addressing, but newer ATA versions also supported LBA addressing. SCSI used LBA addressing. EX2004, pp. 76-79.

c. Files

75. In this Declaration, I apply the plain and ordinary meaning to the term file as controlled by the operating system, as a POSITA would have understood file to mean in the context of the '478 Patent.

76. Petitioners' Expert's Declaration at ¶ 37, citing its exhibit EX1012 (Midgdey), states that "[i]nformation processed by a computer is organized into distinct units called files, each of which contains related data."

77. Petitioners' Expert's Declaration at ¶ 37, citing its exhibit EX1015 (Silberschatz), states that "[t]he operating system provides system calls to create, write, read, reset, and delete files."

78. In other words, a file roughly consists of one or multiple sectors (or blocks) and is created and managed by the file system of the operating system. More precisely, most operating systems commonly allocate storage to files in units called clusters, and a cluster is one or more blocks (or sectors). EX1015, pp. 390- 391

79. Files and sectors are different and should not be conflated.

80. Below are just a few of the main differences between files and sectors.

81. Files are accessible by the end user. A user can create, populate, read, rename, and delete files. In contrast, sectors are not accessible by the end user. A user cannot directly read or write sectors.

82. Files consist of clusters, which consist of sectors. In contrast, sectors are generally the smallest addressable unit in a disk drive.

83. Files are managed by the operating system, specifically the file system. In contrast, sectors are managed by low-level device drivers or BIOS (Basic Input-Output System).

84. Files are managed by the operating system, because files need to be usable with a wide variety of devices. Additionally, file formats allow files to operate on multiple operating systems. For example, text files, Microsoft Word files, Microsoft Excel files, and PDF files are usable on multiple operating systems. In contrast, sectors do not have such requirements, and an individual sector, block, or cluster from one disk drive would have no practical meaning or use to applications or operating systems running on another disk drive.

85. Files are hardware agnostic / independent. File contents are the same regardless of the underlying storage device where the file is stored. In contrast, sectors are hardware dependent, specifically hard disk drive dependent and storage interface dependent.

86. Files are inaccessible without the operating system and file system. In contrast, sectors are accessible through low-level device drivers or BIOS.

87. When the computer first boots, files are inaccessible, whereas sectors are.

88. Files are addressed using file names and paths (folder or directory location). EX1015, pp. 360-370. In contrast, sectors (or blocks) are addressed using CHS or LBA locations.

89. The objective of file-based archive solutions is to backup file system data (the individual files), whereas the objective of sector-based archive solutions is to backup disk drive data (sectors' data).

90. File-based archive solutions can retrieve one or multiple files but not an entire disk drive, whereas sector-based archive solutions can retrieve one or multiple sectors or an entire disk drive. Sector-based archives cannot retrieve individual files.

91. File-based archive solutions can only be used if the operating system is functioning, whereas sector-based archive solutions can be used to restore the entire system, including the operating system.

92. Additionally, in my experience, the engineering skills required to design and implement sector-based archive solutions would have been very different from those required to design and implement file-based archive solutions. For

example, the specific engineering skills for sector-based archive solutions would have included experience with data storage devices, such as disk drives, tape drives, and jukeboxes; experience with programming and interfacing with low-level device drivers and/or BIOS; and experience with data storage interfaces, such as SCSI and IDE. None of these skills are needed for file-based archive solutions.

93. For at least these reasons, sector-based archive solutions are very different from file-based archive solutions.

d. BIOS

94. BIOS (Basic Input / Output System) is software or firmware that provides low-level system services for operating systems and programs – operations such as power-on hardware initialization, system booting, and services for interfacing with hardware and devices (including disk drives). The BIOS services for interfacing with hardware and devices are available to the operating system and application programs.

BIOS provides routines for accessing I/O devices, such as the keyboard, screen, disk drives, and printer, attached to a microcomputer. This involves the following activities:

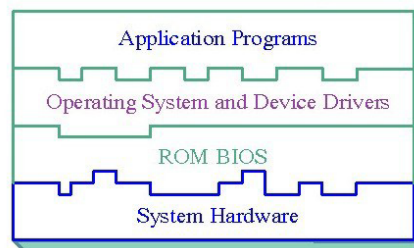
- Checking that the device is ready
- Initiating the data transfer
- Checking for successful transfer
- Performing error correction procedures, if necessary
- Reporting successful completion or permanent error condition

EX2005, p. 196.

A ROM BIOS (Basic Input/Output System) is a set of programs permanently stored in a ROM (Read-Only Memory) chip located on the computer motherboard. These programs micro-manage the hardware devices installed on your computer. When you turn on your computer, the ROM BIOS initializes and tests these devices. During run-time, the ROM BIOS provides the Operating System and application programs with access to these devices. You can also use the BIOS Setup program to change your computer's hardware or behavior.

Software works best when it operates in layers, and the ROM BIOS is the bottom-most software layer in the computer. It functions as the interface between the hardware and the other layers of software, isolating them from the details of how the hardware works. This arrangement enables you to change hardware devices without having to install a new operating system.

The following diagram shows the function of the ROM BIOS as the interface between the hardware and other layers of software:



EX2006, p. 31.

The second task of the ROM BIOS is to provide the Operating System, device drivers, and application programs with access to the system hardware. It performs this task with a set of program routines called BIOS Services, which are loaded into high memory at boot time.

EX2006, p. 32.

95. BIOS deals with disk drive sectors, not files. Files are handled by the operating system and its file system, but not by BIOS.

Most microcomputers are now equipped with disk drives and normally use a disk-based operating system (DOS). DOS routines supplement those in BIOS to provide extended services for using I/O devices. The division of responsibilities between BIOS and DOS for accessing disk storage devices is illustrated by the IBM PC:

- BIOS can read or write to a specified sector, track, and side of a diskette.
- DOS has responsibility for management of space on the diskette, including maintenance of a directory of the files and recording the space that has been allocated.
- DOS provides access to files and their records as logical entities and not by their physical location.

EX2005, p. 197.

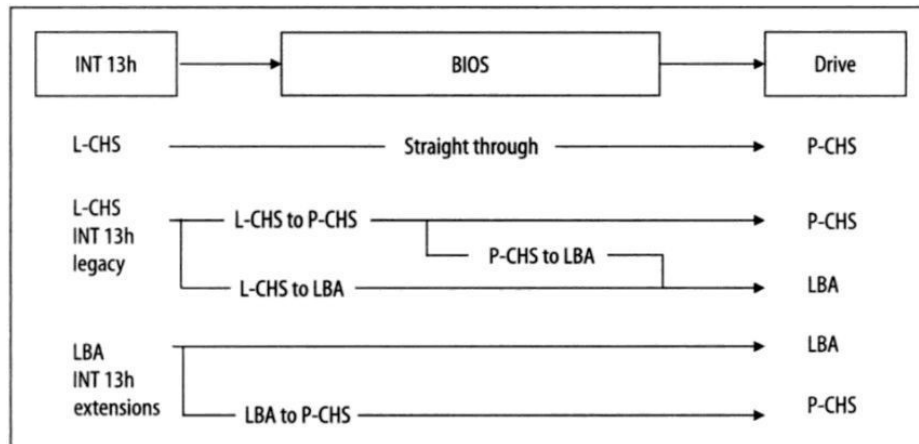


Fig. 5.20 Translation options.

EX2007, p. 132.

96. Exhibit EX2006 p. 80 shows a list of BIOS services. There are no file-based services or functions. All the functions are sector (or block) based. In particular, interrupt 13h is “BIOS Fixed Disk/Diskette Interface.” Exhibit EX2006 pp. 43-51 show the various functions available through interrupt 13h, all of which are sector (or block) based. Exhibit EX2006 p. 33 shows disk drive (AKA fixed disk or hard drive) cylinder-head-sector geometries.

97. In the 1990s and in 2000, operating system software was produced by vendors such as Microsoft and Apple, whereas BIOS software was produced by vendors such as AMI (American Megatrends, Inc.), Phoenix, and Award. EX2004, p. 71.

98. As shown in Exhibit EX2006 p. 32, Phoenix BIOS was used in a variety of operating systems, such as DOS, OS/2, UNIX, Windows 95, and Windows NT. Most other BIOS vendors supported these operating systems.

99. In that timeframe, BIOS functions may be replaced by functions provided by device drivers or memory-resident programs offering similar services to those provided by BIOS.

100. In summary, BIOS, or an equivalent memory-resident program, deals with sector-based or block-based functions but not file-based functions.

III. Claim Construction

101. Based on my review of the Petition's comments pertaining to claim construction, my understanding is that Petition gives claim terms their plain and ordinary meaning as would be understood by a POSITA on the earliest effective filing date, in light of the specification and file history. Petition 12. I address herein how a POSITA would understand certain terms when viewed in light of the disclosure of the '478 Patent.

a. "Searching"

102. The term "searching" is used in the claims and the disclosure of the '478 Patent in its typical manner. It is not specifically defined in the disclosure or the claims. The disclosure the '478 Patent includes two instances of "searching," each of which use the term in a substantially similar manner as in Limitation 1.4. EX1001, 3:31-35 ("searching a first storage location for files responsive to the occurrence of a first event and moving the files..."), 3:43-35 ("the first storage location is searched for an archive file responsive to the occurrence of a first event.").

Neither of these disclosures, nor the surrounding language, give a special meaning to the term “searching” that varies from its plain and ordinary meaning. The term is used similarly in the claims of the ’478 Patent. *See* EX1001, 8:49-50; 9:21-22, 9:36-37, 10:14-15, 10:31-32.

103. I have also reviewed the file history of the ’478 Patent (EX1002), and none of the arguments therein would alter this plain and ordinary meaning for the term “searching” as it is used in the claims of the ’478 Patent.

104. A POSITA would understand the term “search” or “searching” as used in the ’478 Patent claims to have its plain and ordinary meaning, in-line with the following definitions from the Microsoft Press Computer Dictionary:

search¹ . . . The process of seeking a particular file or specific data. A search is carried out by a program through comparison or calculation to determine whether a match to some pattern exists or whether some other criteria have been met. . . .

search² . . . 1. To look for the location of a file. 2. To seek specific data within a file or data structure. . . .

EX2023, p. 424.

105. “[S]earching the first temporary storage location for the archive file” as included in Limitation 1.4 would require seeking the archive file itself, i.e., *specific data*. This would be done by comparison to determine a match with a specified pattern, i.e., seeking the location of the archive file within the first temporary storage location by comparison of the data therein with a pattern for the archive file.

IV. Ground 1 - Long and Kenley

106. Below, I provide a short overview of certain pertinent features of Long and Kenley. In arguing obviousness of the independent Claims of the '478 Patent under Ground 1, Petitioners' Expert Declaration relies primarily on Long, but uses Kenley for three main features in the alleged combination of Kenley and Long: (a) file-based system, (b) creating and storing archive files, and (c) the database structure as Kenley's main features. EX1003 ¶¶ 49-74.

A. Long Overview

107. Long is a sector-based backup and archival system with a mass storage device (e.g., disk drive). Long allocates a partition of the mass storage device for storing backup data and supplemental backup information. To perform backups, Long detects accesses to the mass storage device that would alter its data; upon such detection, Long stores the data to be altered in the audit partition region for potential later restoration.

a. Long is based on disk drive sectors, not files.

108. Unlike the '478 Patent, which is directed to archiving and recovering files, Long is directed to archiving and recovering disk sector data.

109. Petitioners' Expert's Declaration at ¶ 53 alleges that "a POSITA would have known that Long is applicable to files." I disagree.

110. As described earlier in the section titled Disk Drive Sectors versus Files, files and sectors are different. The '478 Patent is directed to, based on, and deals with files, not non-file specific sectors. In contrast, Long is directed to, based on, and deals with just disk drive sectors, not files.

111. The detailed explanation in the following sections show how deeply Long is sector based, not file based, and show how the sector-based approach is intrinsic to, fundamental to, and embedded in Long's invention.

112. Long's invention, operations, events, and functions deal with disk drive sectors, deal with disk drive tracks, and deal with storage devices at low level. Long's implementation is done using BIOS or low-level device driver. Long's invention does not deal with files, file systems, or operating systems.

113. Long is not applicable to files, especially as Petitioners' Expert's Declaration postulates. The combination contemplated in Petitioners' Expert's Declaration relies on Long for most of the '478 Patent claim limitations, other than the database structure. EX1003 ¶¶ 59-95.

114. An understanding of how extensively sector based Long is would undoubtedly deter a POSITA from considering Long for an alleged implementation of the '478 Patent, let alone contemplating modifying Long to make it file based, especially if Long is to be relied on for most of the claim limitations of the '478 Patent, as Petitioners' Expert's Declaration contemplates.

115. A POSITA with the understanding of how extensively sector based Long is would have realized that modifying Long to make it file based, as Petitioners' Expert's Declaration hypothesizes, is tantamount to a new invention.

116. Long does not mention the word file(s). Long does not suggest or imply anything that a POSITA would interpret as files.

117. Long also does not mention operations specific to files, such as "open," "close," "rename," and "delete."

118. Instead, Long uses the word sector(s) 42 times.

119. Below are a few examples of Long's reliance on sectors.

READ SECTORS FROM AUDIT BUFFER" and "WRITE SECTORS TO ARCHIVE MEDIA.

EX1004, FIG. 6 blocks 66 and 68 (emphasis added).

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**.

Id., Table 1 columns 5-6 (emphasis added).

This event occurs when the data on the hard disk(s) is changed. The event contains the data that existed on **one sector** of the disk prior to the write occurring.

Id., 6:46-48 (emphasis added).

[t]he data to be modified on the disk is **read one sector at a time** (block 50). **Each sector is read** from the disk (or from a disk cache, if provided) and is placed in the audit buffer as "before image" data (block 52). If the application requests a write that is more than one sector, **each sector is processed** as

an independent request.

Id., 7:37-43 (emphasis added).

After all of **the sectors of the write request** have been processed, the audit subsystem returns control back to the processing unit 14 and allows it to modify the hard disks 18 with the requested write.

Id., 8:14-17 (emphasis added).

120. Additionally, Petitioners' Expert's Declaration at ¶ 53 concedes that "Long does not specifically consider data stored in files."

b. Long handles sectors containing file data or non-file data but does so on sector basis.

121. Although Long is sector based, Long can process sectors containing file data as well as sectors containing non-file data, but Long does so on sector basis and with no regard to whether the data is file data or non-file data. In other words, Long can process sectors containing data belonging to and organized as files, and Long can process sectors containing data not belonging to and not organized as files, but Long cannot distinguish file from non-file sectors. Long processes file data and non-file data identically, without even knowing if the data is file data and non-file data.

i. Long discloses disk drive tracks.

122. Long also discloses dealing with disk drive tracks, which, as described earlier in the section titled *Disk Drive Sectors*, contain sectors.

When a five second ticker is enabled and the system has determined that it needs to archive, **one track's data** (the size of the track is dependent upon the structure of the archive media) is prepared and then stored on archive media.

EX1004, 8:38-42 (emphasis added).

The archive end marks the end of the **archive track** that is to be written. The **track is then written** to the archive media in block 68.

Id., 9:4-7 (emphasis added).

123. Long's dealing with disk drive tracks further confirms that Long is sector-based.

ii. **Long describes dealing with data storage devices, but not files and not file systems.**

124. Long also describes dealing with data storage devices, but not files and not file systems, and also describes locations on storage devices and states of data storage devices, which are sector-level operations, not file-level operations.

Responsive to a user request, data is restored from the sequence of data transfers in the audit partition region to the respective set of **locations on the storage device** to return the storage device to a **previous state**.

EX1004, 10:26-30 (emphasis added).

The computer system of the present invention includes a mass storage device having a defined audit partition region.

Id., 1:65-67 (emphasis added).

From the history, the user may select a time to which the state of the hard disk will be restored.

Id., 9:24-25 (emphasis added).

The present invention is applicable to any **mass storage media** wherein old data may be overwritten with new data. While the mass media described herein is **described as a “disk”**, it should be noted that other structures, such as a drum structure, could similarly be used without affecting the applicability of the present invention. Further, it should be noted that while the mass storage media typically remains with the computer system 10, the present invention may be used with removable media as well.

Id., 9:54-63 (emphasis added).

a **storage device** comprising ... responsive to a user request, restoring data from the sequence of data transfers in the audit partition region to the respective set of **locations on the storage device** to return the storage device to a **previous state**.

Id., 10:12-30 (emphasis added).

restoring data from said audit partition region to said **storage device** responsive to a user request to return the storage device to a **previous state**.

Id., 11:31-34 (emphasis added).

responsive to a user request, restoring data from the audit partition region to the respective set of **locations on the storage device** to return the storage device to a **previous state**.

Id., 11:34-37 (emphasis added).

responsive to a user input, restoring data from the archival memory to the respective set of **locations on the storage device** to return the state of the storage device to a **previous state**.

Id., 14:19-22 (emphasis added).

125. Long’s descriptions of disk drives and storage devices further confirms that Long is sector-based.

iii. **Long’s operations and events relate to data storage devices, not files.**

126. The ’478 Patent is directed to files and describes operations specific to files (not sectors), such as “open,” “close,” “rename,” and “delete.” In contrast, Long describes hardware and low-level operations and events that relate to data storage devices, not files – operations and events such booting, rebooting, and power failure.

The system events may include, but are not limited to, a **start** condition (for enabling the audit subsystem), **power on** and **power failure, reboot, quiesce** (a user invoked marker for indicating a desired hard disk state) and audit (data changes).

EX1004, 2:8-12 (emphasis added).

The audit buffer 30 is used to record **events such as writes to the hard disk(s), power failure, power on and system reboot.**

Id., 5:26-28 (emphasis added).

The audit partition may be used to store information relating to any number of events. The events supported in the preferred embodiment are described in connection with Table 4. The computer system 10 provides circuitry for generating the appropriate signals responsive to the events. For example, the power supply generates a signal when AC **power is lost**, even though the UPS will continue to operate the computer system 10. Also, the interrupt for a **system reboot** is captured and stored prior to performing the reboot.

Id., 6:27-36 (emphasis added). Long's descriptions of its operations that are specific to disk drives and storage devices (but not operations on files and file systems) further confirms that Long is sector-based.

127. Long also teaches that its rollback utility program reboots the system after restoration is complete. A POSITA would have understood that such approaches are unusable in file-level implementations, which operate at the operating system and file system level.

Typically, **rollback is performed via a utility program** that accesses the on-line system.

...

At that point, the hard disk has been restored to the selected rollback point. After rollback is thus completed, and the user exits the utility, **the utility will reboot the operating system**. After the operating system is loaded, the hard disks will contain the data as of the time of the selected rollback point.

Id., 9:13-14, 9:45-50 (emphasis added).

128. Long teaches operating at the lowest level, namely hardware level. For example, Long teaches hardware-level fault tolerance and hardware-based error detection methods, such as timing checks, ECC, and parity.

The detection of an error or its symptoms during the normal operation of the system is the cornerstone to **fault tolerant** architectures. The component complexity affects the ability to distinguish errors from correct values. Errors occurring in **data-storage components, such as registers and memory**, or during **data transmission via buses or network links**, are more easily detected than errors originating within modules that

generate or transform data. The following methods are used for error detection and/or masking:

1. Module replication for error detection and masking.
2. **Protocol and timing** checks.
3. Error detection and correction codes (**ECC**).
4. Error detection **parity** check codes.
5. Self-checking **logic** (i.e., voting logic).

Id., 3:3-18 (emphasis added).

iv. **Long is implemented as part of the BIOS or as a memory resident program.**

129. Long is implemented either as part of the BIOS or as a memory resident program.

130. Long at 6:60-63 states that “[i]n the preferred embodiment, the audit subsystem **is part of the system BIOS** (Basic Input/Output System), but could also be a **memory-resident program**.” (emphasis added).

131. A memory-resident program (AKA resident program, terminate and stay resident program, or TSR program) is a software program that, when invoked, loads and remains (resides) in the computer’s random access memory so that other the software programs can be called by other programs quickly, without having to load the software program again. EX2024.

132. A POSITA would have also known that, in the context of Long, which is sector (not file) based, a “memory-resident program,” similar to BIOS, deals with

disk drive sectors, not files, and would have known that files are handled by the operating system and file system, not by such memory-resident program.

133. Long's memory-resident program should not be confused with the resident program of the '478 Patent.

134. The '478 Patent defines resident program as an "operating system (OS) or other program that has control over file operations such as "read", "write", "save", "rename", "delete", "copy", "move", "open", "close", etc." EX1001, 4:44-47.

135. In contrast, as described earlier, Long deals with sectors, not files, and does not mention, suggest, or imply file operations such as "open," "close," "rename," or "delete."

136. In summary, BIOS, or an equivalent memory-resident program, deals with sector-based or block-based functions but not file-based functions.

v. Long teaches detecting access requests to alter data on storage device locations (sectors) but not files.

137. Instead of disclosing detecting an instruction to perform an operation on an operating file, Long teaches detecting access requests to alter data on storage device locations (sectors) but not files.

A computer system comprises a storage device comprising a plurality of locations for storing data and having a defined audit partition region comprising one or more of said locations and processing circuitry for **detecting access requests to alter data on respective sets of one or more of locations of the storage device**. Responsive to each access request, the processing

circuitry stores data from said respective set of locations in the audit partition region prior to performing the access request, such that a sequence of data transfers from the storage device is stored in the audit partition region in a known order. Responsive to a user request, data is restored from the sequence of data transfers in the audit partition region to the respective set of locations on the storage device to return the storage device to a previous state.

EX1004, Abstract (emphasis added).

Processing circuitry **detects accesses to the device which would alter data** thereon, and stores the **data to be altered** in the audit partition region for later restoration.

Id., 1:67-2:2 (emphasis added).

processing circuitry for: detecting access requests to alter data on respective sets of one or more of locations of the storage device....

Id., 10:16-19.

processing circuitry for: detecting access requests to alter data on respective sets of one or more of said locations of the storage device....

Id., 11:20-24.

c. A POSITA would have understood Long as teaching away from file system implementation.

138. In light of all the Long teachings described above, a POSITA would have understood Long as teaching away from file system implementation, and a POSITA would have not been motivated to, would have been dissuaded from, and

would not have considered incorporating Long's disk/sector level methods into any file-level implementation (or vice versa).

139. A POSITA would have known that a file-level implementation would necessarily need to be implemented at a much higher level than the "processing circuitry," BIOS, or equivalent memory-resident program taught by Long.

140. A POSITA would have realized that incorporating Long's disk/sector level methods into a file-level implementation would destroy the file system's ability to remain hardware agnostic.

141. A POSITA seeking to improve a file-level implementation would not have considered a reference such as Long, which teaches the lowest possible level implementation, relying on hardware, BIOS, and detection circuitry. Indeed Long, and its constituent components do not deal with or even recognize files or the concept of files.

d. In summary, Long is sector based, not file based, and not applicable to files as contemplated in Petitioners' Expert's Declaration.

142. In summary, unlike the '478 Patent, which is directed to files, Long is directed to sectors. Long is deeply and intrinsically sector based.

143. Long is not applicable to files, especially as Petitioners' Expert's Declaration postulates, which relies on Long extensively and as purely file based.

144. Modifying Long to make it file based would have been a new invention, and not something that would have been obvious to a POSITA.

B. Kenley Overview

145. Kenley is a file-based backup and archival system. Kenley discloses a variation on the concept of hierarchical data storage system for files with a primary storage element, a secondary storage element, and a backing store element, each progressively with greater storage capacity and slower access speed. Data is migrated from to the backing storage elements as fullness threshold is reached or based on time. EX1014, 2:33-68.

C. A POSITA would not have combined Kenley and Long.

146. As discussed above, a POSITA would have understood the significant barriers to any combination of Kenley and Long and, as a result, would not have been motivated to combine the two references.

147. A POSITA would not have had a reasonable expectation of success in combining Kenley and Long for several reasons.

a. Long's teachings specific to sector-level implementation could not be reasonably expected to succeed in file-level implementation and would have contradicted Long's basic teachings.

148. A POSITA in 2000 would have understood that Long's teachings were specific to a sector-level implementation and could not be reasonably expected to

succeed in a file-level implementation, and vice versa. Long was explicitly designed to be sector based, not file based. Long does not mention, suggestion, or imply any file-based application for its invention.

149. A POSITA in 2000 would have understood that modifying Long into a file-based system or combining Long into a file-based system would have contradicted Long's basic and fundamental sector-based teachings, rendering Long inoperable for Long's intended purpose.

150. Furthermore, because of the well-known and fundamental differences between the file layer and sector layer described earlier in the section titled *Computer Data Storage Layers*, a POSITA in 2000 would have understood that extracting functionality from Long into Kenley (or vice versa) would have required substantial redesign, reconstruction, and rewrite of Long's extracted functionality. The well-established computer software hierarchy of layers is intended to separate the system functionality into levels with different capabilities. A POSITA in 2000 would have known that juxtaposing functions from different layers (higher into lower or lower into higher) would have involved significant redesign, reconstruction, and rewrite of the extracted functions.

151. For example, Long (EX1004 at 5:26-30) detects various types of events, such as "writes to the hard disk(s), power failure, power on and system reboot." These events occur at much lower layer than the operating system and file

system layer. Detecting such events is inappropriate and possibly unimplementable at the file level. Additionally, detecting such events at the operating system or file system level would change and negate the principle of operation of the file system and of the layered data storage architecture by embedding low-level functionality into a higher layer.

b. Long is implemented as part of BIOS or resident program, both sector-based, low-level.

152. As described earlier the section titled *Long is implemented as part of the BIOS or as a memory resident program*, Long is implemented as part of BIOS or as resident program. Both are sector-based, low-level, unfitting, and inappropriate for a file-based solution. As described earlier in the section titled *Computer Data Storage Layers*, BIOS and the low-level memory-resident program (device driver) are below the operating system layer and do not deal with files. It would be a violation of computer data storage layered architecture for BIOS and the low-level disk drive device drivers to attempt to access the higher level file system of the operating system.

c. The proposed Kenley-Long combination would have required substantial redesign and reconstruction of Long.

153. A POSITA would have known that modifying Long's system from sector based to file based, as suggested in the Petition, is not trivial. Such endeavor would have entailed almost a complete redesign and reconstruction of Long, almost

certainly necessitating a full rewrite of Long, resulting in significant cost, risk, and time. Beyond such major redesign and reconstruction, Long and Kenley would still need to be integrated, further increasing the cost, risk, and time. The Kenley-Long combination contemplated in the Petitioners' Expert's Declaration requires extensive integration, affecting most aspects of Long. Integration alone (excluding the redesign, reconstruction, and rewrite) is not a trivial task.

The second major topic covered in this chapter is software integration. Integration is the process of combining individual software modules and subsystems into a single working product. Software integration is not a trivial task, especially if the individual software units were developed in different locations or even by different companies separated geographically. This coordinated situation is typical in modern software development.

EX2008, p. 292.

154. Assuming, arguendo, that a POSITA considered Kenley and Long in combination to teach the individual elements of the '478 Patent, given the obvious and expected challenges of combining Kenley-Long as contemplated in Petitioners' Expert's Declaration, the POSITA would not have been motivated to combine Kenley and Long.

d. A POSITA would have recognized that the proposed modifications would have changed the principle of operation of the modified reference.

155. A POSITA would have understood that Long's invention is deeply and intrinsically sector based and would have recognized that the proposed

modifications would have changed the principle of operation of the modified reference.

156. Long is sector based, not file based, and Long's invention is intrinsically sector based. Long's invention, operations, events, and functions deal with disk drive sectors, deal with disk drive tracks, and deal with storage devices at low level. Long's implementation is done using BIOS or low-level device driver. Long's invention does not deal with files, file systems, or operating systems. A POSITA would have realized that modifying Long to make it sector based is complicated, unlikely to succeed, and costly (even today).

157. A POSITA with the understanding of how extensively sector based Long is would have realized that modifying Long to make it file based is tantamount to a new invention (and vice versa).

e. Petitioners' Expert's Declaration at ¶ 25 states that "I understand that in considering obviousness, it is important not to determine obviousness using the benefit of hindsight derived from the descriptions found in the patent being considered."

158. In my opinion, the argument that a POSITA would have been motivated to modify Long to make it file based to be used in combination with Kenley is purely "hindsight derived from the descriptions found in the patent being considered."

D. The Petition and Petitioners' Expert Declaration fail to show that Limitation 1.4 is met by the proposed Kenley-Long combination

159. Limitation 1.4, as labeled by the Petition and Petitioners' Expert Declaration, recites "searching the first temporary storage location for the archive file responsive to the occurrence of a first event." This limitation is not disclosed or suggested in the Long-Kenley Combination.

160. As discussed more fully below, Long does not perform any search of the marker header/trailer/data blocks in the memory buffer when sending that data to the physical hard disk. This is the sequence that is alleged in the Petition and the Petitioners' Expert Declaration to meet Limitation 1.4. The Petition and Petitioners' Expert Declaration also point to Long's "semaphoring," which also does not meet Limitation 1.4, because this semaphoring / blocking would only indicate the presence of some data in the memory buffer. The semaphoring would not involve a search for any particular data within the memory buffer.

a. Long's flushing of the memory buffer does not disclose a search for any alleged archive file.

161. The Petition and Petitioners' Expert Declaration cite to two disclosures of Long for Limitation 1.4's "first event" – "(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 (cited by Petition at 31; Ex 1003 ¶ 66).

162. Petitioners' Expert Declaration then states that "[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." EX1003 ¶ 66 (citing EX1001, 5:46-49 (giving non-limiting examples of "event[s]" as messages from resident program(s))). Long's BIOS is not a "resident program" (as described in the '478 Patent) as Petitioners' Expert Declaration appears to claim. As discussed above, Long's BIOS functions at a lower level than the operating system and file system. Long's BIOS is not a "resident program" as discussed in the disclosure of the '478 Patent, which is explicitly defined as "an operating system (OS) or other program that has control over file operations such as 'read', 'write', 'save', 'rename', 'delete', 'copy', 'move', 'open', 'close', etc." EX1001, 4:44-47. Long's BIOS does not fit this definition, as it does not have control over file operations.

163. Petitioners' Expert Declaration then states that

Either event triggers the audit subsystem searching the memory buffer for the data 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.").

EX1003 ¶ 66. I disagree that there would be any searching in Long's moving the data in this instance. As discussed above in the claim construction section, searching would require seeking the location of the archive file within the first temporary storage location by comparison of the data therein with a pattern for the

archive file. The quote from Long states that the memory buffer containing the audit markers is not “searched,” but it is rather “flushed.” In Long, when the memory of the memory buffer is needed by another part of the system, the memory buffer is “flushed” to the physical disk. Flushing entails copying data to another location, in this case the physical disk, and then erasing the data from the original location, in this case Long’s memory buffer. This can, and likely would, be done as a block copy in Long, which is a copy of a set of memory without searching through the memory. The audit header tracks the beginning and the end of the Long memory buffer, which is stored contiguously in memory. This enables a block copy of the full memory buffer from the first memory location through the last without any search of the markers / data within the buffer.

164. Petitioners’ Expert Declaration then argues that “[t]o flush or move the memory buffers data, the system must search the memory buffer for the data to locate the data; absent such searching, the system would neither know what it is moving nor when the move has completed.” This does not at all follow from flushing or moving the memory buffer. As apparently acknowledged by Petitioners’ Expert Declaration, the memory buffer of Long holds multiple markers. EX1003 ¶¶ 67-68. The memory buffer’s holding multiple markers is also reflected, for example, in the fact that the memory buffer of Long has its own audit header. EX1004, 7:47-52 (when the memory buffer is full or the memory is otherwise required by the system,

“the audit subsystem writes the data stored in the memory buffer to the end of the audit buffer 30 on disk (block 56) and updates the audit header (in the memory buffer) to reflect the changes (block 58).”). As stated by Petitioners’ Expert Declaration, the audit header stores information regarding markers. EX1003 ¶ 68. However, importantly, the audit header of Long stores only information on the first and the last markers in the sequence of markers. Below is Table 1 from Long, which details the data stored in Long’s Audit Header, as highlighted by Petitioners’ Expert Declaration in paragraph 77 thereof:

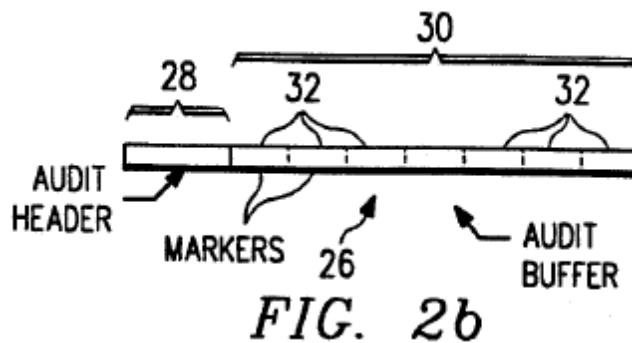
TABLE 1

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

Audit Header Fields	
Data Label	Data Description
	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.

As can be seen in Table 1 of Long, as far as information regarding markers, only data regarding the first and the last markers is stored in the Audit Header of Long. (FirstMarkerLoc, FirstMarkerOffset, FirstMarkerSector, LastMarker, LastMarkerLoc, LastMarkerOffset). The second, third, fourth, etc., markers of Long are not tracked by the Audit Header. The Audit Header and markers are stored in a sequence, as shown in Figure 2b of Long:



Long's Figure 2b is described as showing the audit partition on the disk (EX1004, 4:40-42), but it is equally applicable to the memory buffer that stores the same sort of markers and audit headers. When the memory buffer is flushed, or copied to the disk, the system does not need to search the contents of the memory buffer at all, as the Audit Header stores the information on the first and the last markers, with the remaining markers stored continuously in memory. The data from the start to the end of the buffer is copied together to the physical disk and does not need to be searched by the Long system at all. Petitioners' Expert Declaration is incorrect that the contents of the memory buffer would need to be searched in order to flush the

memory buffer. The system continues to track the beginning and the end of the buffer and needs only this information to enable the markers within the memory buffer to be moved and appended to the end of the set of markers already in the audit partition on the physical disk. The audit header of the memory buffer is reset and the information regarding the last marker (and the overall length of the memory buffer, which can be determined from the locations of the first and last markers) will be used to update the Audit Header of the audit partition.

165. In sum, the contents of the Long memory buffer (the markers stored therein) would not be searched in response to the two events that are alleged in paragraph 66 of Petitioners' Expert Declaration to constitute a "first event."

166. Petitioners' Expert Declaration, in describing the markers and the audit header of Long in paragraphs 67-68, states that this information is "used to aid locating the archived data when searching the memory buffer for the archived data" responsive the two events. This statement assumes that the search is actually performed. As outlined above, however, the memory buffer's markers are not actually searched in response to the events provided in Petitioners' Expert Declaration (and the same events are given in the Petition).

b. Long’s semaphoring the memory buffer does not indicate a search for the alleged archive files.

167. Petitioners’ Expert Declaration and the Petition also cite to a Long disclosure regarding the memory buffer being semaphored. EX1003 ¶ 69; Petition at 33 (both quoting EX1004, 8:18-20). As stated by Petitioners’ Expert Declaration, a semaphore “indicates the presence of the data within the memory buffer to other processes.” EX1003 ¶ 69 (citing EX1019, p. 354 (defining “semaphore”)). As defined in Exhibit 1019, a semaphore is a “signal” or a “flag variable—an indicator” used to govern access to a shared system resource. EX1019, p. 354. A semaphore as used in the context of Long – having the memory buffer “semaphored” – simply means that there is a signal or flag-variable that indicates whether the memory buffer is editable or locked.

168. Even assuming that the memory buffer is only locked when data is present in the memory buffer, the presence of a semaphore would merely indicate that data is present, and nothing about the contents of the data. Furthermore, nothing indicates that the data in the memory buffer would be searched for individual header information responsive to any event in this instance. The semaphore would merely block the ability to write data to the memory buffer until the prior data in the memory buffer is flushed, the buffer would then be empty, and the semaphore signal would

change from blocking writing to allowing writing. This semaphore process does not involve searching the memory buffer for any particular data.

c. The proposed combination with Kenley does not cure the above deficiencies of Long.

169. After the above discussion of Long, Petitioners' Expert Declaration simply states that using these operations in a file-based system, such as Kenley, would "result in the audit subsystem searching the memory buffer (first temporary storage location) for the copied/archive file responsive to a first event." EX1003 ¶ 70. But this conclusion does not at all follow. As discussed above, the cited portions of Long do not disclose or suggest searching the memory buffer for any particular data; hence, merely porting these Long mechanisms over to Kenley – even assuming that this would be possible – does not disclose or suggest "searching the first temporary storage location for the archive file," as recited in Limitation 1.4.

170. The same arguments apply to Limitations 8.4, 9.4, 10.4, and 11.4 (EX1003 ¶ 113), and Limitations 8.4, 9.4, 10.4, and 11.4 are not disclosed or suggested by Kenley-Long for the same reasons. Claims 2-6 depend from claim 1 and are patentable for the same reasons.

E. The Petition and Petitioners' Expert Declaration fail to show that Limitation 1.6 is met by the proposed Kenley-Long combination.

171. Limitation 1.6, as labeled by the Petition and Petitioners' Expert Declaration, recites "after storing the archive file in the first temporary storage

location, updating a database to indicate that the archive file is located in the first temporary storage location.” This limitation is not disclosed or suggested in the Long-Kenley Combination.

172. As discussed more fully below, the Petition and Petitioners’ Expert Declaration allege that Kenley-Long purportedly teaches Limitation 1.6 based on the concept of “stor[ing] Long’s audit header information in Kenley’s relational database.” However, as discussed above in paragraph 163, Long’s audit header does not store location information about all of Long’s markers. Long’s audit header only stores such information about the first and the last marker.

173. Petitioners’ Expert Declaration starts its discussion of Limitation 1.6 with characterizations of Kenley and its database. EX1003 ¶ 76. Petitioners’ Expert Declaration points to a system catalog that is maintained by Kenley. *Id.* (citing EX1014, 3:17-28). Keeping a database and “maintaining it,” of course, does not teach storing the location of any archive files. Petitioners’ Expert Declaration then points to a disclosure of Kenley that stores “backup catalogs (i.e., lists of all files copied to specific backup savesets).” *Id.* (citing EX1014, 6:35-40). Lists of files and “savesets” merely indicate which files are stored in a particular group but do not disclose the location of any particular archive file. Petitioners’ Expert Declaration then points to a disclosure of Kenley describing a “database 14 a catalog of all files backed up.” *Id.* (citing EX1014, 8:26-29). Finally, for Kenley citations for

Limitation 1.6, Petitioners' Expert Declaration points to a disclosure in the background regarding storing data on offline media, such as tapes. *Id.* (citing EX1014, 2:6-14). Petitioners' Expert Declaration argues that this disclosure somehow shows failure to track locations as a problem that Kenley sought to solve, but the disclosure actually shows a problem with management of offline media but nothing about file location. The Kenley citation certainly does not disclose updating a database to indicate that an archive file is located in the first temporary storage location, as recited in Limitation 1.6. None of the Kenley disclosures in paragraph 76 of Petitioners' Expert Declaration show Kenley teaching Limitation 1.6. To address this Kenley deficiency, Petitioners' Expert Declaration relies on Long, even though Long does not cure the deficiency.

174. In paragraph 77, Petitioners' Expert Declaration argues that Long "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 (first temporary storage location)." EX1003 ¶ 77 (citing EX1004, 7:49-53). Petitioners' Expert Declaration then points to Long's disclosure that 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.* (citing EX1004, 4:44-47, 4:57-60, 2:5-8). Petitioners' Expert Declaration then points to "a number of fields" in Long's audit header that purportedly "specify the location of data managed by the audit

subsystem,” including “CacheLoc” (specifying the location of the audit buffer itself), “BufLoc” (specifying the physical location of the beginning of the audit buffer), and “CLoc” (specifying the physical location of the last data in the buffer). *Id.* (citing EX1004, 4:61-5:24 (Table 1)).

175. As with my discussion above in paragraph 163, the location information stored in Long’s audit header only concerns the beginning and the end of the buffer, and nothing about the markers in between.

176. Petitioners’ Expert Declaration then states that “Long discloses ‘construct[ing] the audit header’ for a marker after ‘[s]torage of the data in the memory buffer’ in the marker.” EX1003 ¶ 77 (citing EX1004, 7:59-68). However, this statement mischaracterizes Long. Long’s system does not construct an “audit header” for a marker. The Long disclosure that follows states that the audit header is created from “the type of marker (corresponding to the type of event), the markers’ CRC, the current system time, and the location that the sector data is from on the hard disk.” EX1004, 7:65-68. This information is not part of Long’s “audit header,” but rather part of Long’s *Marker Header*. EX1004, 5:35-53 (Table 2). A Long audit header is not created for each marker with location information stored therein. Long’s audit header is only used once at the front of a buffer (*see* EX1004, Fig. 2b), and the audit header stores only data for the first and the last marker in its buffer.

177. Petitioners’ Expert Declaration then argues that “when the data is moved from the memory buffer (temporary storage) to the audit buffer, the audit header is updated ‘to reflect the changes.’” EX1003 ¶ 77 (citing EX1004, 7:47-53). Again, the audit header is updated only for the last marker that is added to an audit buffer; the audit header does not keep track of the location of any other markers other than the first and the last markers. Petitioners’ Expert Declaration then states that “[c]onstant 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. This statement is false. Long’s audit header tracks only the first and the last marker in an audit buffer, and any purported “constant updating” merely keeps track of the beginning and the end of the audit buffer and does not indicate the location of the remaining data within the memory buffer associated with the audit header.

178. Petitioners’ Expert Declaration then states that “[t]hus, 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 relevant audit-header information taught by Long, was stored in the relational database catalog taught by Kenley.” EX1003 ¶ 78. In other words, using Long’s audit header information in Kenley’s

database is the basis for the combination's alleged teaching of Limitation 1.6. However, as discussed above, Long's audit header does not store the location of the markers in the audit buffer; only the first and the last marker locations are stored. Hence, Long's audit header does not store the location of the purported "archive files" in the Kenley-Long combination using Kenley's database.

179. Petitioners' Expert Declaration concludes on this point by stating that "as with other elements, when used to implement a file-based system such as Kenley's, these database updates taught by Long would indicate that the copied file (archive file) is located in the memory buffer (first temporary storage location)." EX1003 ¶ 79. Once again, the purported "database updates" taught by Long do not track the location of the markers in an audit buffer. The audit header merely tracks the start and the end of the audit buffer (the first and the last marker), so the alleged combination would not allow for updating the Kenley database to track the location of "archive files." Long's audit headers keep track of the beginning and the end of their buffers, but do not track the location of the full set of markers.

180. In summary, the Kenley-Long combination does not disclose or suggest "after storing the archive file in the first temporary storage location, updating a database to indicate that the archive file is located in the first temporary storage location" as recited in Limitation 1.6. Claims 2-6 depend from claim 1 and are patentable for the same reasons.

F. The Petition and Petitioners' Expert Declaration fail to show that Limitation 1.9 is met by the proposed Kenley-Long combination.

181. For the same reasons as discussed above in Section E in relation to Limitation 1.6, the Kenley-Long combination does not disclose or suggest “updating the database to indicate the archive file is located in the intermediate storage location” as recited in Limitation 1.9.

182. Petitioners' Expert Declaration uses the same logic for Limitation 1.9 as for Limitation 1.6.

183. Petitioners' Expert Declaration starts its Limitation 1.9 argument by stating that “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 ‘determ[ing] the audit header (in the memory buffer) to reflect the changes.’” EX1003 ¶ 89 (citing EX1004, 7:47-53; FIG. 5). This statement merely shows that Long updates the audit header in the memory buffer (which the Petition and Petitioners' Expert Declaration map to the first temporary storage location), when data is moved to physical disk to indicate that the memory buffer is empty. The statement says nothing about the audit buffer on the disk, which is the purported intermediate storage location of the combination.

184. Petitioners’ Expert Declaration then states that “[t]he audit header is updated to indicate ‘[t]he physical location of the beginning of the audit buffer on disk.’” EX1003 ¶ 89 (citing EX1004, 4:61-5:24 (Table 1) (defining “BufLoc”)). This is false. Although there is a BufLoc variable in the audit header of the disk that tracks the beginning of the audit buffer, the BufLoc variable is not updated when data is added to the end of the audit buffer on the disk when data is added to the disk’s audit buffer (the alleged intermediate location) from the memory buffer (the alleged first temporary storage Location). This is made clear by the quoted portion of Long in paragraph 89 of Petitioners’ Expert Declaration – data is written “to the end of the audit buffer 30 on disk.” EX1003 ¶ 89 (citing EX1004, 7:47-53). Because data is added at the end, the “BufLoc” that tracks the beginning would not be updated.

185. Petitioners’ Expert Declaration then states that “[a]s 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.’” EX1003 ¶ 90 (citing EX1014, 6:35-40). As discussed above in paragraph 173, the Kenley disclosure related to lists of files copied to specific backup savesets does not teach tracking the location of particular files, alleged as “archive files.”

186. Petitioners’ Expert Declaration then states that Kenley-Long would meet Limitation 1.9 for the same reason it purportedly meets Limitation 1.6. As

discussed above in Section E, Kenley-Long fails to meet Limitation 1.6 and, for the same reasons, it fails to meet Limitation 1.9.

187. The Kenley-Long combination does not disclose or suggest “updating the database to indicate the archive file is located in the intermediate storage location” as recited in Limitation 1.9. Claims 2-6 depend from claim 1 and are patentable for the same reasons.

G. The Petition and Petitioners’ Expert Declaration fail to show that Limitation 1.10 is met by the proposed Kenley-Long combination.

188. For the same reasons as discussed above in Sections E and F in relation to Limitations 1.6 and 1.9, the Kenley-Long combination does not disclose or suggest “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” as recited in Limitation 1.10. Petitioners’ Expert Declaration uses the same logic for Limitation 1.10 as for Limitations 1.6 and 1.9.

189. Petitioners’ Expert Declaration states that “Long teaches updating its audit header to indicate that the copied (archived) data is located in the removable archive media (second storage location)” after the data has been moved. EX1003 ¶ 92 (citing EX1004, 9:7-9; *id.*, Fig. 6). The cited portion of Long merely shows the audit header of the disk removing reference to whatever is moved at the end of the

audit buffer. The citation does not show any database update to indicate that any data is located in the archive media.

190. Petitioners' Expert Declaration cites to Long's audit header as "includ[ing] fields specifying the location of data in the archive media that are 'used to manage the ... archive media.'" EX1003 ¶ 93 (citing EX1004, 4:44-47). While Long in column 4 mentions that the audit header is used to manage the archive media, the Long citation related to the audit header or the fields pointed to in paragraph 93 of Petitioners' Expert Declaration does not disclose or suggest using Long's audit header to keep track of the location of the markers of Long in the archive disk. Petitioners' Expert Declaration points to the "next Archive disk sequence number to use" and the "last archive disk that contains a user quiescence marker." EX1003 ¶ 93 (citing EX1004, 4:61-5:24 (Table 1)). Neither of these fields, a sequence number or the last disk that contains a user quiescence marker, indicates the location of any markers in Long. None of these disclose or suggest that Long updates any of its audit headers to indicate the archive file is located in the second storage location, as recited in Limitation 1.10.

191. Petitioners' Expert Declaration then states that "[a]s 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.'" EX1003 ¶ 94 (citing EX1014, 6:35-40). As discussed above in paragraph 173 and 185, the Kenley

disclosure related to lists of files copied to specific backup savesets does not disclose tracking the location of particular files, alleged as “archive files.”

192. Petitioners’ Expert Declaration then states that Kenley-Long would meet Limitation 1.10 for the same reason it purportedly meets Limitations 1.6 and 1.9. As discussed above in Sections E and F, Kenley-Long fails to meet Limitations 1.6 and 1.9 and, for the same reasons, Kenley-Long fails to meet Limitation 1.10.

193. In summary, the Kenley-Long combination does not disclose or suggest “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” as recited in Limitation 1.10. Claims 2-6 depend from claim 1 and are patentable for the same reasons.

V. Ground 2 - Cabrera, Schneider, and Whiting

A. The Petition fails to show that Limitation 1.4 is disclosed or suggested by Ground 2’s proposed combination.

194. The proposed combination of Cabrera, Schneider, and Whiting does not disclose or suggest “searching the first temporary storage location for the archive file responsive to the occurrence of a first event” as recited in Limitation 1.4.

a. The cited Cabrera disclosures do not disclose searching for the archive file responsive to a first event.

195. Petitioners’ Expert Declaration cites to three aspects of Cabrera for this limitation, none of which discloses or suggests the limitation.

196. First, Petitioners' Expert Declaration cites to language that Cabrera monitors the "amount of data in the staging area ... in order to identify when a backup or archive session should be initiated." EX1003 ¶ 144 (citing EX1005, 3:45-47). This citation does not pertain to searching for any archive file in the temporary storage location. Instead, the citation simply describes monitoring the total amount of data in the alleged temporary storage location.

197. Petitioners' Expert Declaration then cites to a general disclosure in Cabrera that the "'event monitor 152' ... '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.'" EX1003 ¶ 144 (citing EX1005, 13:1-10). This statement mentions potential for monitoring and triggering events, but does not pertain to the temporary storage location and does not pertain to searching the temporary storage location for the archive file.

198. Petitioners' Expert Declaration then states the following:

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 would have understood that evaluating whether to append more data to the staging file necessarily requires searching the staging area for the staging file, and that this searching step would occur responsive to the triggering event.

EX1003 ¶ 144. This allegation rests on the presumption that the claimed

"searching ... for the archive file" is met by Cabrera's purported "searching the

staging area for the staging file.” EX1003 ¶ 144. Petitioners’ Expert Declaration equates Cabrera’s “staging file” with the claimed “archive file.” As I discuss below, however, the two are not the same, and Cabrera does not disclose or suggest searching the staging area for the alleged archive file.

b. The staging file of Cabrera does not correspond to the claimed archive file

199. The staging file of Cabrera is a conglomeration of data. It is not the individual archive file recited in the challenged claim. Cabrera, from its most general description of its systems, discloses the staging of “data” before data is transferred to remote storage. EX1005, 1:9-11 (“the present invention relates to Systems and methods for temporarily storing or staging data prior to its transfer to remote storage.”). Cabrera’s systems require access to files “twenty-four hours a day, seven days a week,” such that one cannot “prevent access to a particular file during the archive or backup procedure.” EX1005, 1:64-2:3. In systems with such a requirement, “creating backup or archive copies of information” can be challenging, but one method employed by Cabrera is to “duplicate the information that will be backed up or archived and ‘stage’ the information in a temporary storage area.” EX1005, 2:3-9. The “*information*” so staged “may then be copied from the staging area and sent to backup or archive Storage.” EX1005, 2:9-10.

200. In other words, Cabrera requires access to files at all times, such that access to files cannot be blocked during the backup process, so files are first copied to a staging area and subsequently copied to archival / backup. The staging area contains a set of data, which constitutes copies of many files. The background of Cabrera discusses utilizing archive media of a defined size, and collecting sufficient information to completely fill this archive media before archiving the data. EX1005, 2:35-40. This collection is the staging area. EX1005, 2:41-43.

201. Cabrera's description of maintaining data in the staging area and appending data when more is available is as follows:

The amount of data in the staging area may be monitored in order to identify when a backup or archive session should be initiated. In the alternative, the time since last backup or archive may be monitored and a session initiated when a particular time has elapsed. If additional data becomes available in local storage prior to the time that an archive or backup session is initiated, such data can be [sic.] appended to the data already stored in the staging area.

EX1005, 3:45-53. The above paragraph describes "data" in the staging area – not a particular file that is the staging file. Cabrera's appending data if more is available in the local storage area before the archive file discloses adding "data" to the conglomeration of data within the staging area. The Cabrera paragraph does not disclose any particular operating file that is stored as any sort of "staging file" in the staging area. In this context, the data refers to the data that has been staged,

waiting either for sufficient data in the staging area or sufficient time to elapse in order for an archive process should be started.

- c. **The “staging file” is only disclosed in conjunction with Cabrera’s Figure 5, and does not differ from the “staging area” – a conglomeration of data from the local disk.**

202. Cabrera’s Figure 5 is discussed in the specification from column 10, line 37 and continuing on to column 12, line 17. In the discussion of Figure 5, Cabrera’s description of the staging area shifts from discussing it as an “area” with “data” to a “staging file.” This description is not a separate embodiment or any actual shift in the underlying mechanism, but merely a change in nomenclature. This “staging file” of Cabrera is a conglomeration of data, not any specific file that has been copied for back up. This “staging file” is data waiting to be backed-up to archival media, like an optical disk.

203. To make it clear that there is no change in mechanism from the remainder of Cabrera, the staging area discussion of Figure 5 uses the same language as earlier in the Cabrera specification:

In FIG. 5, the method begins with decision block 124 which identifies whether sufficient data resides in local storage for staging to a staging area. If sufficient data does not reside in local storage for staging, the system waits for a given period of time illustrated in FIG. 5 by time delay 126, and then rechecks the amount of data in local storage.

EX1005, 10:39-45.

204. Multiple subsequent mentions of the staging area in the Cabrera discussion of Figure 5 use the term “staging file.” *See, e.g.*, EX1005, 10:56-61 (after triggering event, “execution proceeds to Step 128 where data is copied from local storage to the staging file.”). This disclosure confirms that data, conglomerated, is copied to the staging area, referred to as the staging file. Data, *i.e.*, a conglomeration, is copied to *the* staging file. The staging file is a collection of data and not, for example, “a” staging file that corresponds to any particular file from the local disk.

205. The primary disclosure of Cabrera relied upon by Petitioners’ Expert Declaration as well as the Petition and the Institution Decision for Limitation 1.4 is as follows:

Thus, decision block 130 determines whether it is time to establish a remote storage connection. If not, then execution proceeds to decision block 138 where a determination is made as to whether more data should be appended to the staging file. This determination may be made based on any triggering event, as previously explained in conjunction with decision block 124 and decision block 130. **If more data that should be appended to the staging file exists, then the data is appended in step 140.** In either case, execution proceeds back to decision block 130 in order to wait for the occurrence of the second triggering event that will initiate connection to remote storage.

EX1005, 11:20-31 (emphasis added). As Petitioners’ Expert Declaration argued, “A POSITA would have understood that evaluating whether to append more data to the staging file necessarily requires searching the staging area for the staging file, and that this searching step would occur responsive to the triggering event.”

EX1003 ¶144. The Institution Decision relied on the same logic. Institution Decision at 37-38.

206. However, even accepting, *arguendo*, that the logic as true and that appending data to a file requires searching for that file, Cabrera still does not disclose that the staging file is the archive file, such that it would have to be searched. Cabrera discloses appending to the *staging file*, which is the whole set of data that is waiting to be backed up to remote storage, to determine if more data from local storage should be appended to the staging file before beginning the backup process. When data is appended to the “staging file,” it is not appended to the “archive file” that is derived from an original “operating file” as required by Limitation 1.4.

207. Limitation 1.4 requires searching for the archive file in response to a first event. Cabrera’s disclosure of appending data to the staging file does not entail searching for any particular archive file (corresponding to an initial operating file) within Cabrera’s staging area / staging file.

208. Therefore, Cabrera does not disclose or suggest “searching the first temporary storage location for *the archive file* responsive to the occurrence of a first event” as recited in Limitation 1.4.

209. The same arguments apply to Limitations 8.4, 9.4, 10.4, and 11.4 (EX1003 ¶ 155), and Limitations 8.4, 9.4, 10.4, and 11.4 are not disclosed or

suggested by Ground 2's combination for the same reasons. Claims 2-6 depend from claim 1 and are patentable for the same reasons.

B. The Petition fails to show that Limitation 1.5 is disclosed or suggested by Cabrera-Schneider-Whiting

210. Limitation 1.5 is “moving the archive file to a second storage location responsive to a second event, the second storage location being a permanent storage location.” The proposed combination of Cabrera, Schneider, and Whiting fails to disclose or suggest Limitation 1.5.

a. Ground 2 uses Cabrera's intermediate system to meet Limitation 1.8.

211. It is important to note that, in order to meet other limitations of the challenged claims, Petitioners' Expert Declaration utilizes the limited disclosure of Cabrera regarding “an intermediate system or subsystem.” *See* EX1003 ¶149 (regarding Limitation 1.8). This Cabrera disclosure is used to meet Limitation 1.8's recitation of moving the archive file from the temporary storage location to “an intermediate storage location.”

212. Ground 2, as mapped by Petitioners' Expert Declaration and the Petition, utilizes the following disclosure of Cabrera:

By way of example, and not limitation, in FIG. 2 such means is illustrated by arrows 78, 80, and 82 which illustrate data being moved from sparse file 76 and delivered to remote storage communication infrastructure 84. Such a means may be implemented by any mechanism capable of retrieving data from

sparse file 76 and either directly delivering the data to remote storage 62 or to an intermediate system or subsystem which will, in turn, deliver the appropriate data to remote storage.

EX1005, 7:52-60 (cited by EX1003 ¶ 149). Petitioners' Expert Declaration and the Petition also cite to Figure 2 and column 8, lines 5 to 11, which discuss the remote storage infrastructure, without reference to the intermediate system or subsystem.

b. Limitation 1.5 requires moving the archive file to a second / permanent storage location responsive to a second event.

213. In mapping Limitation 1.5 in Ground 2, Petitioners' Expert Declaration points to Cabrera's teaching that "[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 (cited by EX1003 ¶ 145). Petitioners' Expert Declaration also points to potential triggers that could constitute this alleged second event.

214. But, even if, *arguendo*, this mapping of the second event and the second storage location is correct, this second event could only trigger the movement of the archive file from the staging area / staging file to remote storage in Cabrera's system when there is no intermediate system or subsystem. The relied-upon disclosure of Cabrera for Limitation 1.5 ignores Cabrera's intermediate system or subsystem, which Ground 2 requires to meet other limitations of the challenged claims.

215. Nothing in the Petition, the Petitioners' expert declaration, or Cabrera itself describes what the mechanism is for moving data from Cabrera's intermediate

location to the remote storage. There is no disclosure of any trigger or alleged second event in Cabrera that could meet the requirements of Limitation 1.5, “moving the archive file to a second [/permanent] storage location responsive to a second event.”

c. Cabrera is silent on the when or what might trigger moving data from the intermediate system to remote storage.

216. Cabrera provides scant description of the intermediate system. Cabrera mentions intermediate system in three parts. The first mention, referenced above, is in column 7, where Cabrera merely states that the mechanism for moving data could be “any mechanism capable of retrieving data from sparse file 76 and either directly delivering the data to remote storage 62 or to an intermediate system or subsystem which will, in turn, deliver the appropriate data to remote storage.” EX1005, 7:56-60. This disclosure does not indicate when data would move from the intermediate system or subsystem to remote storage or, importantly for purposes of Limitation 1.5, what, if any, alleged “second event” would trigger such a move. Petitioners’ Expert Declaration and the Petition offer nothing regarding these aspects of the proposed combination of Ground 2.

217. The other mentions of the intermediate system in Cabrera describe that the mechanism to transfer data “may be nothing more than writing data to a local disk or other storage device or this may require transferring data over various networks or via various computer systems or other intermediate devices to remote

storage.” EX1005, 11:49-55. And “remote storage 158” may comprise a wide variety of storage mechanisms including, among other things, “a remote storage device accessed via an intermediary or other intermediary computer device.” EX1005, 13:62-67 (as corrected by Certificate of Correction).

218. None of these brief disclosures regarding Cabrera’s intermediate system gives any detail about how or when data would be moved from the intermediate system to the remote storage. For purposes of Limitation 1.5, none of the disclosures provide any detail about what, if anything, might trigger such a move such that it could be alleged as the “second event” of Limitation 1.5. Petitioners’ Expert Declaration and the Petition say nothing on this point.

219. Therefore, the proposed combination of Ground 2 – Cabrera, Schneider, and Whiting – fails to disclose or suggest “moving the archive file to a second storage location responsive to a second event, the second storage location being a permanent storage location” as recited in Limitation 1.5, and Ground 2 fails for this additional reason.

220. The same arguments apply to Limitations 8.5, 9.5, 10.5, and 11.5 (EX1003 ¶ 155), and Limitations 8.5, 9.5, 10.5, and 11.5 are not disclosed or suggested by Ground 2’s combination for the same reasons. Claims 2-6 depend from claim 1 and are patentable for the same reasons.

C. A POSITA would not have combined Cabrera and Schneider as in Petitioners' Expert Declaration because doing so would not work and would fundamentally alter the operation of Cabrera.

a. Cabrera is periodic in nature.

221. Cabrera relates to periodic backing up of data, where data is sent from the local storage area to the staging area for the purpose of backing up the data “when a defined amount of data has been stored locally or when a particular time has elapsed.” EX1005, 3:40-41. “The amount of data in the staging area may be monitored in order to identify when a backup or archive session should be initiated. In the alternative, the time since last backup or archive may be monitored and a session initiated when a particular time has elapsed.” EX1005, 3:45-50. This periodic nature is important in the functioning of Cabrera.

222. The proposed combination of Cabrera and Schneider with Whiting is based on the argument that “it would have been obvious to use a file manipulation event as taught by Schneider as a triggering event to initiate the backup-archival process taught by Cabrera.” (EX1003 ¶ 139). I disagree that it would have been obvious to take the periodic back-up process of Cabrera, and fundamentally change how it operates to trigger a back-up on a file manipulation event. Making this change would fundamentally alter the operation of Cabrera.

223. Combining Cabrera with Schneider as proposed in Petitioners' Expert Declaration would eliminate steps 124 and 126 from Cabrera. EX1005, 10:45-49

(“decision block 124 and time delay 126 illustrate a mechanism whereby a system periodically checks to see if sufficient data resides in local storage to be staged to the staging area.”). Aside from sufficient data, Cabrera discloses “the triggering event for staging data to the staging area would be the expiration of an elapsed time rather than the accumulation of an amount of data.” EX1005, 10:53-55.

224. Cabrera’s backup process is periodic, and, in my opinion, triggering backup based on a file manipulation event would not have been obvious to a POSITA.

b. The proposed combination would not work with the system of Cabrera.

225. Cabrera only discloses moving data that has already been written by its data producers. The proposed combination would intercept a disk write before such data is written, and thus is fundamentally incompatible with the underlying operations of Cabrera.

226. Petitioners’ Expert Declaration states that “Cabrera teaches a variety of events that trigger” the backup method and that Cabrera discloses “[a] first triggering event for copying data to the staging area includes ‘the accumulation of a given amount of data, the expiration of a given time delay, the receipt of a command to stage data, or any other triggering event.’” EX1003 ¶ 139 (quoting EX1005, 10:53-60).

227. Petitioners' Expert Declaration uses Schneider's teaching that "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." EX1003 ¶ 138 (citing EX1020, 12:5-8). This "intercepting the write" is basically inserted into Cabrera's "any other triggering event" for Limitation 1.2. Schneider's "intercepting the write," however, would not work as Cabrera's triggering event for copying data from local storage to the staging area, because Cabrera discloses only backing up data that has already been written by Cabrera's data producers.

228. Petitioners' Expert Declaration uses Cabrera's disclosure from column 6, lines 58-62 as creating the "data that is to be backed up or archived" for Limitation 1.2. EX1003 ¶ 137. This is the alleged operating file of the challenged claims, which corresponds to Cabrera's Data File (64). Cabrera's Data File (64) is described in the context of Figure 2 and the related discussion in Cabrera. Importantly, Cabrera's Data File has already been created / written before Cabrera's back-up procedure is started. Cabrera discloses

In FIG.2, one or more data producers, such as data producer 60, creates data that is to be backed up or archived to a backup or archive storage device, such as remote storage 62. The data produced by data producer 60 is stored in a local storage medium, such as the hard disk for the local computer system. In FIG. 2, data producer 60 is illustrated as storing data in data file 64.

EX1005, 6:58-67. This is the method that Cabrera uses to create data. No other method is disclosed in Cabrera for creating data.

229. Cabrera discloses moving data files to the staging area:

When a first event occurs, indicating that data local storage area should be transferred to a staging area, archive system 66 will move an appropriate amount of data from the local storage area, such as data file 64, to a staging storage area which is adapted for temporarily storing the data.

EX1005, 7:3-7. Cabrera further discloses “the present invention may comprise means for moving data from a local storage area used for data storage by a data producing service to a staging storage area used for temporarily staging the data.”

EX1005, 7:7-11. Copying data or direct memory transfer are example processes for moving data to the staging area. EX1005, 7:11-20.

230. Cabrera backs up data produced by the data producing service, namely data after it has been produced. Cabrera can only copy data that has already been written to the local disk. A POSITA would recognize that “‘intercepting the write’ to trigger copying the file on the disk *before completing the write*” (EX1003 ¶ 137) would not work in the system of Cabrera because Cabrera copies/transfers data to the staging area only after the data has been written / produced by the data producer and not before a write has been executed.

231. Figure 2 illustrates the point, and is copied below:

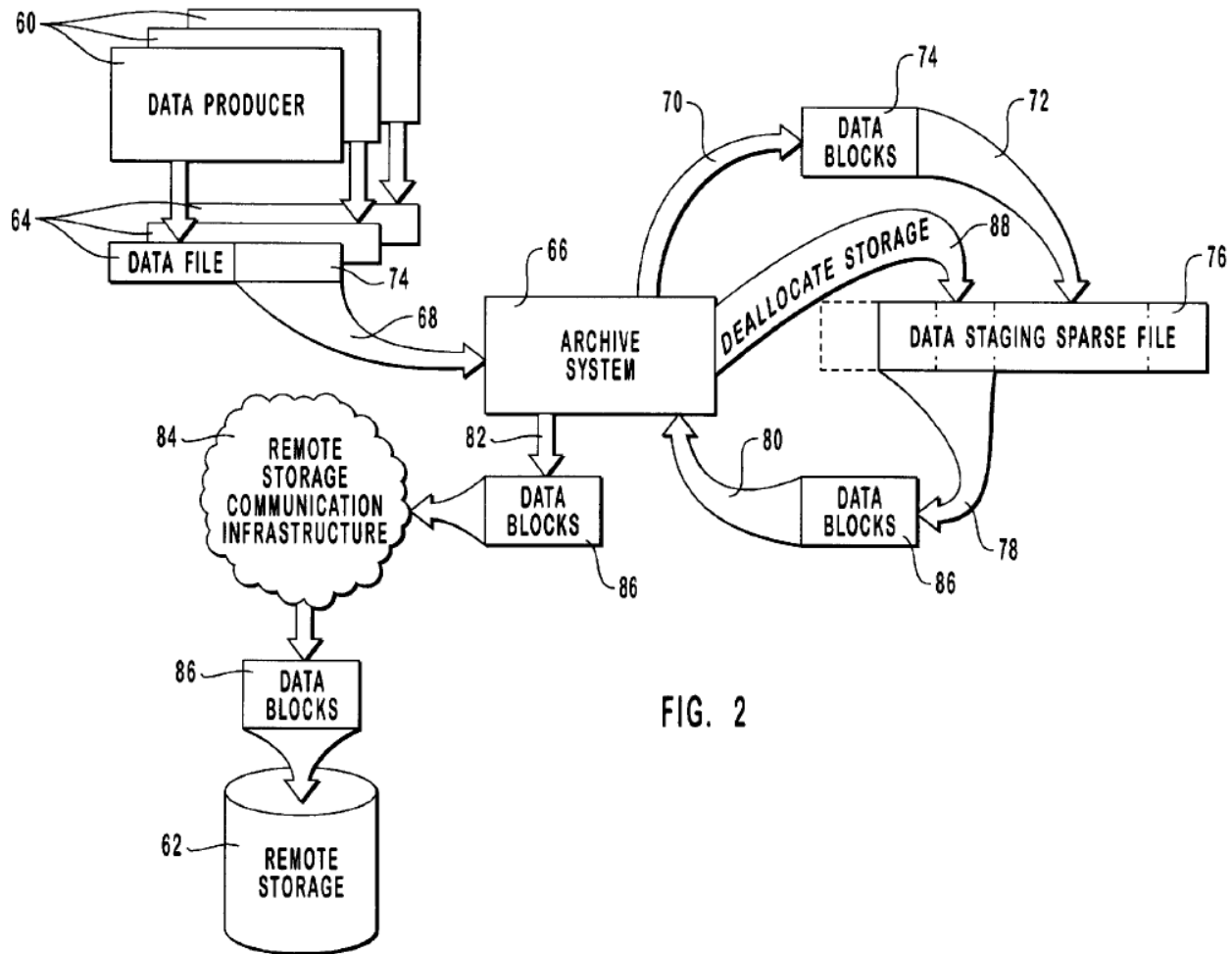


FIG. 2

EX1005, Fig. 2. As seen in the Figure, Data Producer 60 produces Data File 64.

The downward facing arrows from Data Producer 60 to Data File 64 represent the production / writing / editing of the Data Files in Cabrera. Arrow 68 is the “means for moving data” from to the archive system. EX1005, 7:16-20. This means for moving data, 68, comes after the downward facing arrows from the Data Producer to the Data File. This shows that Cabrera’s archive system is not interposed between the Data Producer and the Data File such that data could be sent to the

archive system before it is written. In Cabrera, data must be produced before it is backed up to the staging area.

D. Whiting does not disclose or suggest updating a database to reflect storage of an archive file in multiple locations and a POSITA would not combine Whiting with Schneider and Cabrera.

232. The Petition and Petitioners' Expert Declaration use Whiting in Ground 2's combination to address the database limitations (Limitations 1.6, 1.9, and 1.10).

233. Regarding motivation to combine, Petitioners' Expert Declaration does not specify how to combine Whiting into the combination, and simply argues for using Whiting's database to improve Cabrera's ability to track data. EX1003 ¶ 135.

234. Whiting is added into the combination of Ground 2 without full explanation. Whiting has a backup directory file that purportedly stores location data of backed-up files in Whiting. There is not a thorough discussion of Whiting or how or why Whiting would fit into the combination of Ground 2 with the other two references. Whiting is used merely to meet the database features of Limitations 1.6, 1.9, and 1.10.

235. The thrust of Whiting, however, is not a multi-location backup system. The backup data in Whiting is all stored in one backup disk. *See, e.g.*, EX1006, Fig 1, 7:13-17 ("As shown in FIG. 2. each node is assigned two directories, a user directory and a system directory, on the backup storage means 101, which is contained in the disk volumes of the network file server 100."). There is only one

network file server 100 with a single backup storage means 101 in Whiting. Whiting does not disclose moving backup data between multiple locations nor updating a database to indicate that an archive file is located in each of multiple locations. This is because the disclosure of Whiting is primarily concerned with just one storage location.

236. There is one paragraph of Whiting indicating that in one embodiment, there could be hierarchical storage management (HSM), where “files that have not been accessed for a long time are migrated from disk to a secondary storage means, such as tape or optical disk” to save on storage costs. *See* EX1006, 6:52-7:6. No details are provided by Whiting about how this HSM would work or how it would affect Whiting’s backup directory. Petitioners’ Expert Declaration and the Petitioner are unclear about whether and how Whiting would track locations of data when it is stored in a secondary storage means.

237. In Petitioners’ Expert Declaration’s discussion of Limitation 1.6, there is only one quote from Whiting: “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.’” EX1003 ¶ 146 (citing EX1006, 8:45-50).

238. The Petitioners’ Expert Declaration discussion for Limitations 1.9 and 1.10 simply points to the discussion for Limitation 1.6. EX1003 ¶ 150. This is

because, contrary to Limitations 1.9 and 1.10, which recite updating the database to indicate that the archive file is in various storage locations, Whiting does not disclose multiple backup data storage locations and does not disclose tracking the movement of backup data between such locations.

VII. Conclusion

239. For the foregoing reasons, in my opinion, independent claims 1, 8, 9, 10, and 11 and dependent claims 2-6 of the '478 Patent are not obvious in view of Kenley and Long and are not obvious in view of Cabrera, Schneider, and Whiting.

240. I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct.

August 2, 2024

Date



Zaydoon ("Jay") Jawadi