

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

DROPBOX, INC. AND MICROSOFT CORPORATION,
Petitioners,

v.

DATANET, LLC,
Patent Owner.

IPR2024-00078
U.S. Patent No. 8,473,478

REPLY DECLARATION OF DAVID MAIER

DROPBOX EXHIBIT 1024 <i>Dropbox v. Datanet</i> IPR2024-00078
--

TABLE OF CONTENTS

	Page
EXHIBIT LIST	4
I. INTRODUCTION	6
II. BACKGROUND AND QUALIFICATIONS	6
III. INFORMATION CONSIDERED	6
IV. RELEVANT LEGAL STANDARDS	7
V. CLAIM CONSTRUCTION	7
A. Mr. Jawadi’s Proposed Construction Of “Searching” Ignores Intrinsic Evidence And Imports New Limitations	7
VI. GROUND 1: CLAIMS 1-6 AND 8-11 ARE OBVIOUS IN VIEW OF KENLEY AND LONG.....	12
A. Long’s Modification-Driven Backup Method Improved, And Was Compatible With, Kenley’s File-Based Backup System	12
1. Mr. Jawadi Ignored Long’s Modification Request.....	12
2. Mr. Jawadi Did Not Dispute That Schneider-US Already Confirmed Long’s Applicability To Kenley	13
3. There Is No Fundamental Incompatibility Between File-Based And Sector-Based Techniques	18
4. Long Does Not Teach Away From Combination With File-Based Systems	20
5. Combining Kenley And Long Does Not Destroy The Principle Of Operation Of Kenley But Instead Improves It	22
B. Kenley-Long Renders Element 1.4 Obvious	24
1. Kenley-Long Searches For The Archive File In Order To Move It	24

C.	Kenley-Long Renders Elements 1.6, 1.9, And 1.10 Obvious.....	31
1.	Kenley-Long Updates A Database To Indicate The Location Of The Archive File In Order To Track The Archive File	31
VII.	GROUND 2: CLAIMS 1-6 AND 8-11 ARE OBVIOUS IN VIEW OF CABRERA, SCHNEIDER, AND WHITING.....	34
A.	A POSITA Would Have Found The Cabrera-Schneider-Whiting Combination Obvious.....	35
1.	Cabrera-Schneider-Whiting Improves Upon The Operation Of Cabrera By Preserving Files Before Modification	35
2.	A POSITA Would Have Known That Cabrera-Schneider-Whiting Would Work As Proposed.....	37
3.	Whiting’s Express Teachings Emphasize Obviousness Of The “Database” Claim Elements.....	40
B.	Cabrera-Schneider-Whiting Evaluates The Staging Area, And Thus Searches For The Archive File	41
C.	Cabrera-Schneider-Whiting Moves The Archive File To A Permanent Storage Location Responsive To A Second Event, And Moves The Archive File To An Intermediate Storage Location Along The Way	42

EXHIBIT LIST

LIST OF NEWLY FILED EXHIBITS

Exhibits concurrently filed with Petitioners' Reply to Patent Owner Response:

No.	Description
1024	N/A
1025	U.S. Patent No. 5,907,672 (“ Matze ”)
1026	Deposition Transcript of Zaydoon Jawadi, dated October 8, 2024 (“ Jawadi Tr. ”)
1027	Microsoft Press Computer Dictionary (1994) (excerpts)
1028	U.S. Provisional Application No. 60/234,211 (“ ’211 Provisional ”)

LIST OF PREVIOUSLY FILED EXHIBITS

No.	Description
1001	U.S. Patent No. 8,473,478 (“ ’478 patent ”)
1002	File History of U.S. Patent No. 8,473,478 (“ ’478 FH ”)
1003	Declaration of David Maier, dated October 31, 2023 (“ Maier Decl. ”)
1004	U.S. Patent No. 5,325,519 (“ Long ”)
1005	U.S. Patent No. 5,953,729 (“ Cabrera ”)
1006	U.S. Patent No. 5,778,395 (“ Whiting ”)
1007	Marc Farley, <i>Building Storage Networks</i> (2000) (excerpts) (“ Farley ”)
1008	Helen Custer, <i>Inside the Windows NT File System</i> (1994) (excerpts) (“ Custer ”)
1009	Newton’s Telecom Dictionary (2000) (excerpts)

1010	U.S. Patent No. 5,860,124 (“ Matthews ”)
1011	Ann Chervenak, et al., <i>Protecting File Systems: A Survey of Backup Techniques</i> (“ Chervenak ”)
1012	U.S. Patent No. 5,485,606 (“ Midgley ”)
1013	U.S. Patent No. 5,745,313 (“ Sliger ”)
1014	U.S. Patent No. 5,276,867 (“ Kenley ”)
1015	Abraham Silberschatz, et al., <i>Operating System Concepts</i> (3d ed 1991) (“ Silberschatz ”)
1016	U.S. Patent No. 6,016,553 (“ Schneider-US ”)
1017	Joost S.M. Verhofstad, <i>Recovery Techniques for Database Systems</i> , Computing Surveys (June 1978) (“ Verhofstad ”)
1018	U.S. Patent No. 5,771,354 (“ Crawford ”)
1019	Microsoft Press Computer Dictionary (1994) (excerpts)
1020	PCT Publication No. WO 99/12101 (“ Schneider ”)
1021	U.S. Patent No. 5,684,991 (“ Malcolm ”)
1022	Russell Sandberg, et al., <i>Design and Implementation of the Sun Network Filesystem</i> , in <i>Innovations in Internetworking</i> (Craig Partridge ed., Artech House 1988) (excerpts) (“ Sandberg ”)
1023	List of Challenged Claims

I, David Maier, do hereby declare as follows:

I. INTRODUCTION

1. I have been retained as an independent expert on behalf of Dropbox, Inc. and Microsoft Corporation in connection with the above-captioned Petition for *Inter Partes* Review (“IPR”) to provide my analyses and opinions on certain technical issues related to U.S. Patent No. 8,473,478 (hereinafter “the ’478 patent”). To that end, I prepared a Declaration (EX1003) (“First Declaration”) in support of the Petition.

2. I have prepared this Reply Declaration in connection with Petitioners’ Reply in support of their IPR of the ’478 patent, which I understand is to be filed concurrently with this declaration. I respond herein to certain points raised in the Declaration of Mr. Zaydoon Jawadi (EX2020) filed by Patent Owner Datanet, LLC.

II. BACKGROUND AND QUALIFICATIONS

4. My background, qualifications, and compensation for this matter are set forth in my First Declaration, which I incorporate herein by reference.

III. INFORMATION CONSIDERED

5. In preparation for this declaration, I have considered the materials discussed in this declaration, including, for example, the ’478 patent, the prosecution history of the ’478 patent, various background articles and materials referenced in this declaration, and the prior art references identified in this declaration. I have also

reviewed the expert declaration of Mr. Zaydoon Jawadi and all of the cited exhibits contained therein, submitted on behalf of the Patent Owner. I have also considered the technical arguments in the Patent Owner Response filed by the Patent Owner. I have also reviewed the transcript of the deposition of Mr. Jawadi, and all of the exhibits used therein. I have also considered the Board's decision instituting this proceeding. In addition, my opinions are further based on my education, training, experience, and knowledge in the relevant field.

6. I reserve the right to supplement or amend this declaration if additional facts and information that affect my opinions become available.

IV. RELEVANT LEGAL STANDARDS

7. I am not an attorney and offer no legal opinions. For the purposes of this Reply Declaration, I have been informed about certain aspects of the law that are relevant to my analysis. My understanding of the law is summarized in my First Declaration.

V. CLAIM CONSTRUCTION

A. Mr. Jawadi's Proposed Construction Of "Searching" Ignores Intrinsic Evidence And Imports New Limitations

8. My understanding of how claim terms should be given their plain and ordinary meaning to a person of ordinary skill in the art (POSITA) in view of the specification and file history is consistent with how Mr. Jawadi described his

understanding of claim construction. EX1003, ¶33; EX2020, ¶26. But while Mr. Jawadi purports to construe the term “searching” in element 1.4 as having its plain and ordinary meaning (EX2020, ¶¶102-104), Mr. Jawadi ignores how a POSITA would understand the term after reading the entire ’478 specification and the file history (i.e., intrinsic evidence), and instead turns to extrinsic evidence to provide a narrower construction of the term. Specifically, Mr. Jawadi uses a dictionary definition of “search” to suggest that this plain and ordinary meaning requires “seeking a particular file or specific data,” and that “[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.” EX2020, ¶104 (quoting EX2023, 424). I disagree with this dictionary-based construction of the term “searching” as used in the challenged claims, because the plain and ordinary meaning of the term in view of the specification and file history is broader than the offered definitions. Further, construing the term “searching” according to these dictionary definitions would amount to importing new limitations into the claims that are not disclosed or suggested in the ’478 patent.

9. Regarding the use of the term “searching” in the ’478 patent’s specification, Mr. Jawadi points to the only two instances of “searching” and concludes that each “use the term in a substantially similar manner” to the claim and that the surrounding disclosure does not “give a special meaning to the term

‘searching’ that varies from its plain and ordinary meaning.” EX2020, ¶102. The first instance cited by Mr. Jawadi broadly discloses, as part of “a method for moving files from a first storage location to a second storage location,” the two steps of “searching a first storage location for files responsive to the occurrence of a first event and moving the files from the first storage location to the second storage location responsive to a second event.” EX1001, 3:29-35 (cited by EX2020, ¶102). Similarly, in the second instance cited by Mr. Jawadi, “the first storage location is searched for an archive file responsive to the occurrence of a first event.” EX1001, 3:44-45 (cited by EX2020, ¶102). Both instances indicate a meaning of “searching” that is broader than the dictionary-based definition offered by Mr. Jawadi, because the disclosures indicate searching for “any files” or “an archive file” in order to move them out of the storage location accordingly, without any suggestion of seeking a *particular* file or *specific* data.

10. This broader ordinary meaning is consistent with the other description in the ’478 patent’s specification informing the plain and ordinary meaning of “searching,” which was not referenced or otherwise addressed by Mr. Jawadi. According to the disclosure in the ’478 patent relevant to searching, “smart data management block 15 regularly examines input buffer 20 for the presence of archive files.” EX1001, 5:39-41. Similarly, “[i]n step 100 of FIG. 2A, smart data manager 15 examines input buffer 20 to determine whether any archive files are stored

therein.” EX1001, 6:21-23. Then, “[i]f no archive files are present, smart data manager 15 rests idle until the next event occurs,” but “[i]f archive files are detected, in step 105, smart data manager 15 updates database 25 to indicate the location of the archive files; that is, to indicate that the archive files are resident in the input buffer 20.” EX1001, 6:23-27. According to FIG. 2A of the ’478 patent, “SDM examines it’s INPUT BUFFER to see if any files reside there,” with the flowchart branching off depending on whether files are present or not present:

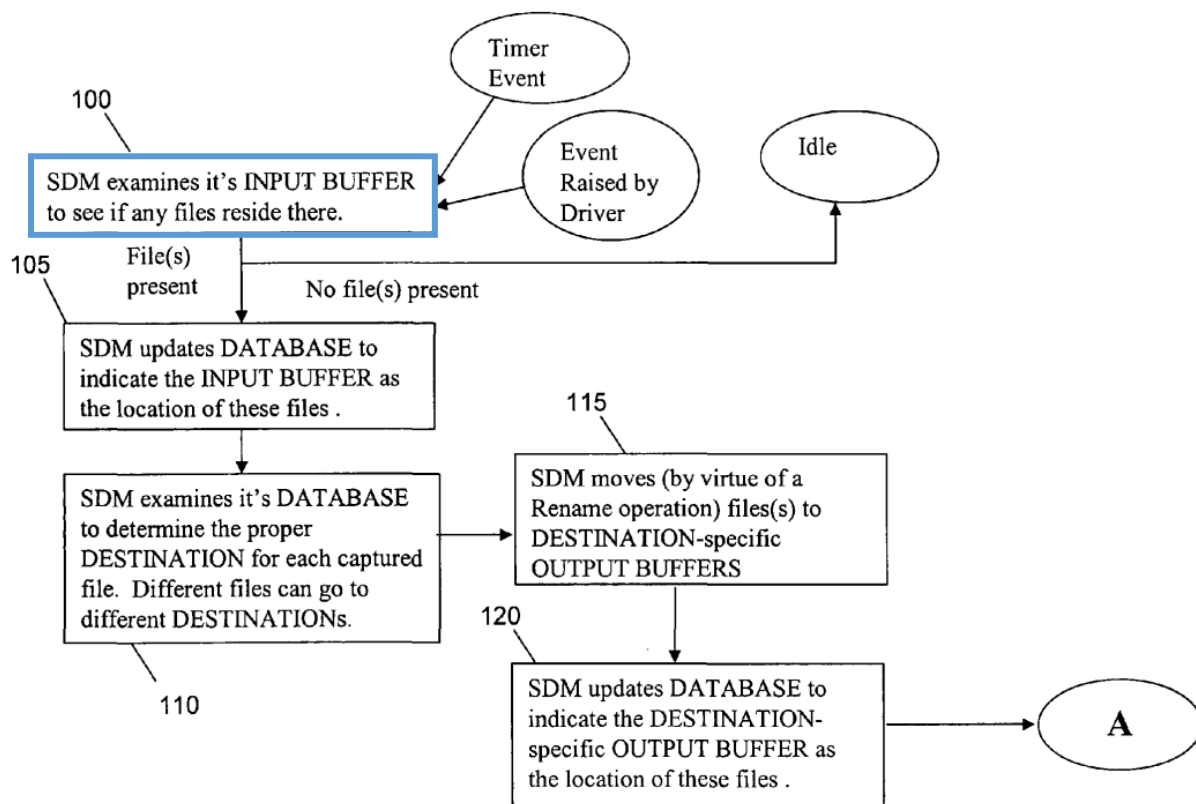


FIGURE 2A

EX1001, FIG. 2A (annotations added). A POSITA, after reading the entire '478 patent, would understand that the plain and ordinary meaning of “searching” as used in the challenged claims to be a general examination or determination of whether *any* files are located in the first temporary storage location.

11. During prosecution of the '478 patent, the applicant cited these “examining” passages in an appeal brief as providing “[e]xemplary support” for “searching the first temporary storage location for the archive file responsive to the occurrence of a first event.” EX1002, 147/744 (citing EX1002, 726 (¶43, ll. 6-11), 728 (¶45)). Mr. Jawadi allegedly “reviewed the file history of the '478 Patent,” but did not acknowledge these passages. EX2020, ¶103.

12. A proper construction of the plain and ordinary meaning of “searching” in the '478 patent must consider this disclosure and how a POSITA would have understood it. Rather than consider the intrinsic evidence, Mr. Jawadi’s construction ignored it.

13. The dictionary-based definitions offered by Mr. Jawadi are narrower than the plain and ordinary meaning of the claim term in view of the intrinsic record, and consequently would require additional steps that are not taught or suggested. For example, if the plain and ordinary meaning of the term “searching” required “seeking a particular file or specific data” as Mr. Jawadi suggests, how would the *particular* file or *specific* data be sought? What is being compared or calculated “to

determine whether a match to some pattern exists”? Mr. Jawadi provides an unclear explanation, using a blend of the two definitions: “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.” EX2020, ¶105. Mr. Jawadi offers no reasoning and cites no support for this explanation. It is unclear what “pattern for the archive file” would be compared to data within the first temporary storage location, or where the method obtains the pattern, as the ’478 patent discloses no such comparison nor any patterns.

**VI. GROUND 1: CLAIMS 1-6 AND 8-11
ARE OBVIOUS IN VIEW OF KENLEY AND LONG**

**A. Long’s Modification-Driven Backup Method Improved,
And Was Compatible With, Kenley’s File-Based Backup System**

14. Mr. Jawadi did not dispute key teachings and evidence from the prior art. Instead, Mr. Jawadi posed various arguments that the combination of Kenley and Long was not obvious, all based on the premise that Long’s modification-driven backup method was not compatible with Kenley’s file-based backup system. Each argument fails.

**1. Mr. Jawadi Ignored
Long’s Modification Request**

15. Mr. Jawadi’s declaration is completely silent with regard to the “modification request” in Long. When asked about the modification request at

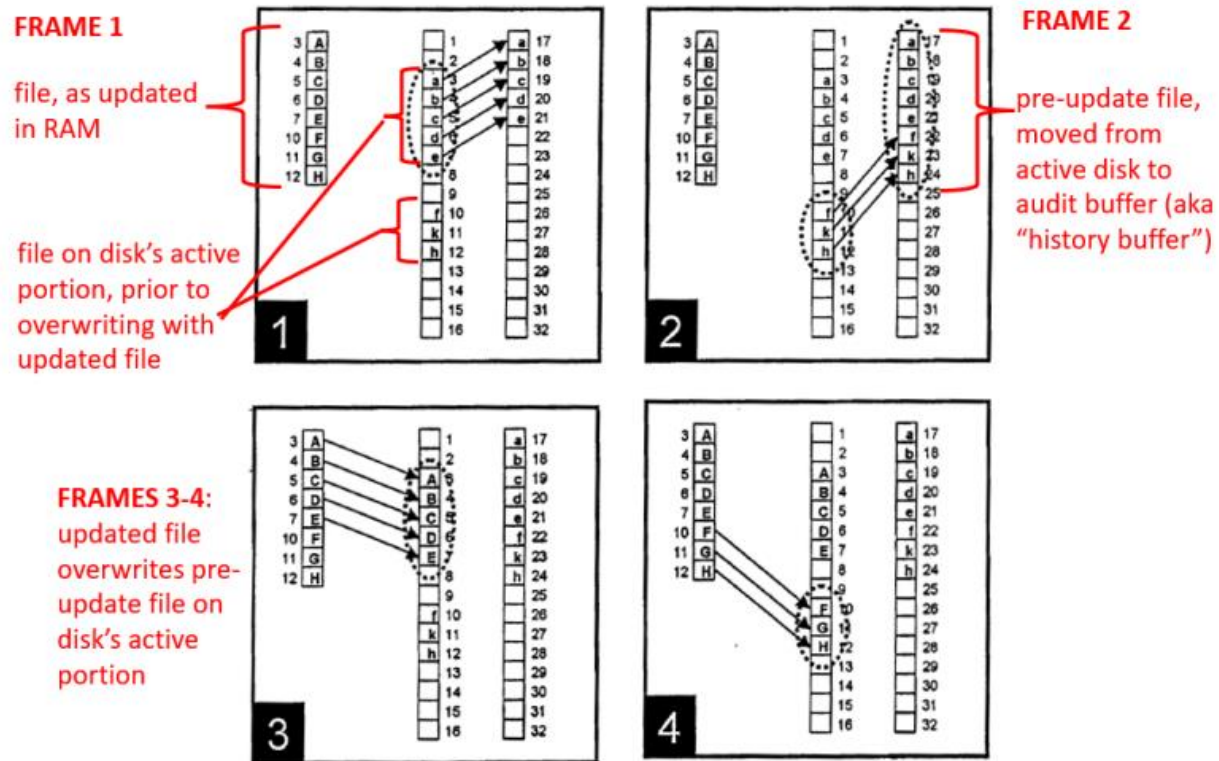
deposition, Mr. Jawadi did not recognize it or understand what was being discussed. EX1026, 203:5-24. Long describes initiating a backup–archiving process in response to an application requesting to modify data on a hard disk. EX1004, 7:28-29. Nothing about this teaching is restricted to sectors, and a POSITA would have easily implemented it in a file-based system. See EX1003, ¶¶49-58. Mr. Jawadi’s assertion that “modifying Long’s system from sector based to file based ... would have entailed almost a complete redesign and reconstruction of Long” (EX2020, ¶153) completely ignores the knowledge and ordinary creativity of a POSITA with respect to the teachings of Kenley and Long, as well as the actual proposed combination where Kenley’s file-based backup–archival system is combined with Long’s teaching of initiating backup–archiving in response to a “modification request,” i.e., a request from an application to modify data (e.g., a file) on a hard disk. See, e.g., EX1003, ¶¶49-51, 58.

**2. Mr. Jawadi Did Not Dispute That Schneider-US
Already Confirmed Long’s Applicability To Kenley**

16. Mr. Jawadi does not dispute or address Schneider-US, which I cited as evidence confirming “[t]he reasonable expectation of success, and overall obviousness, of implementing Long’s technique for a file-based system such as Kenley’s.” EX1003, ¶57. Instead, Mr. Jawadi argues that “sector-based archive solutions are different and incompatible with file-based archive solutions.” EX2020, ¶63. Mr. Jawadi also argues that “Long is not applicable to files” (EX2020, ¶113),

apparently because Long does not use the word “file” (EX2020, ¶116), despite also admitting that “[a]lthough Long is sector based, Long can process sectors containing file data” (EX2020, ¶121).

17. But as I explained in my First Declaration, Schneider-US “expressly describes use of Long’s audit buffer technique for saving a file’s data before the file is altered,” and thus evidences the applicability of Long’s teachings to a file-based system. EX1003, ¶57 (citing EX1016, 46:36-47:8). Schneider-US describes the technique in Long as the “Move Method,” one of “five fundamental methods of recording the original state of information prior to its being altered.” EX1016, 6:33-35, 6:42, 4:27-38 (citing and summarizing Long), 8:7-39 (explaining the “Move Method” and noting that “[t]he basic elements of the Move Method are described in the ’519 patent” (Long, EX1004)), 8:40-52 (discussing use of the Move Method with *files*), 46:63-47:9 (describing how the Move Method first copies a *file* about to be overwritten into a history buffer). I provided an annotated copy of Schneider-US’s FIG. 38, which “illustrates the Move Method” applied to a file. EX1016, 46:63; EX1003, ¶57.



EX1016, FIG. 38 (annotated based on description at EX1016, 46:36-47:9); EX1003, ¶157. According to the description in Schneider-US, FIG. 38 shows four frames, each frame depicting three columns of boxes, each box containing “a value for a page in the file” and labeled with “disk locations (location keys) assigned by the [operating system].” EX1016, 46:38-43. The left column “represents a file” (EX1016, 46:40-41), the middle column represents an “OS-visible area” of a disk (EX1016, 46:65-66), and the right column represents a partitioned portion of the disk reserved for auditing (EX1016, 46:66-47:1). These “frames occur after the user continues working,” that is, after the user commands the writing of the file to disk. EX1016, 53-56. Schneider-US explains that “[i]n Frame 1 the file is overwritten” as

represented by the left column, “but before the hard disk is modified, the affected pages are moved into the history buffer” as shown by the copying of the boxes from the middle column to the right column in Frames 1-2. EX1016, 47:1-3. Schneider notes that “Frame 1 shows the reading of the data about to be overwritten and where it is eventually placed,” but “for the moment the data goes into a buffer” (i.e., temporary storage location) not shown. EX1016, 47:3-5. Then, with “both areas[] having been loaded into a buffer” (temporary storage location), the file is “written to the disk-based history buffer” (second storage location) in the right column as depicted in Frames 1-2. EX1016, 47:6-8. The file on disk is then overwritten as shown in Frames 3-4. EX1016, 47:8-9.

18. As I explained in my First Declaration, Schneider-US’s discussion of Long’s technique confirms a POSITA would have naturally and easily implemented Long’s method for file-based systems. EX1003, ¶57. Mr. Jawadi’s allegations that “Long is not applicable to files” and that “sector-based archive solutions are different and incompatible with file-based archive solutions” fail to address the already-known applicability of Long’s technique to file-based systems as plainly described and even illustrated in Schneider-US, which was published January 18, 2000, prior to the alleged invention date of the ’478 patent.

19. Prior art reference Schneider (EX1020) discussed in Ground 2 also confirms the applicability of Long to files and a file-based backup–archival system.

Similar to Schneider-US, but with a slightly different overall disclosure, Schneider also discusses implementing the “Move Method” from Long as a “fundamental method[] of recording the original state of information prior to its being altered.” EX1020, 34:12-18, 6:23-29 (discussing Long). Schneider explicitly describes how Long’s move method can be implemented “at the file level,” where “[s]uch an implementation could save an entire prior file state just before a file is opened and the intention to modify is clear.” EX1020, 24:1-2, 24:20-24; *see also* EX1020, 102:25-103:22 (describing the Move Method applied to a file). This further confirms Long’s applicability to files and Kenley’s file-based backup–archival system.

20. Further, despite providing a sector-based embodiment, Long more broadly claimed a method that did not invoke sectors. For example, claim 31 of Long recited a method comprising, *inter alia*, “detecting access requests to alter data on respective sets of one or more said locations of the storage device,” and “responsive to each access request, storing data from said set of locations in the audit partition region prior to performing the access request.” EX1004, 12:19-37 (claim 31); EX2022, 49:2-9 (discussing claim 31 and noting that “Long, itself, claims to be broader than simply monitoring sector operations”); *see also* EX1004, Abstract (summarizing Long’s method without restricting it to sectors); Petition, 19 (quoting EX1004, Abstract).

21. I testified at deposition that a POSITA would have understood that Long's methods would "be equally applicable to managing file page data." EX2022, 47:23-48:2. Regarding whether Long describes detecting an operating system's command itself, I explained: "I believe Long would detect file-based operations, file-modification operations emanating from the operating system or a user program because those file modifications ultimately get down to the level where you are writing disk." EX2022, 48:8-13. "I think for the purposes of this combination, the important thing is that it's possible to recognize[,] it would not be hard for a POSITA to come up with a number of ways to recognize that file modification requests were emanating from application programs or the operating system." EX2022, 48:14-20. This realization is especially apparent given that Long expressly defines a modification request as "a request *from an application* to modify the contents of the hard disk." EX1004, 7:28-29 (emphasis added).

**3. There Is No Fundamental Incompatibility
Between File-Based And Sector-Based Techniques**

22. Mr. Jawadi argues that "techniques and solutions for managing data at the file system level are rarely applicable to the disk/sector level, and vice versa," simply because "file systems and disk sectors exist at separate conceptual levels." EX2020, ¶¶65-66. But Mr. Jawadi does not explain why the techniques are "rarely applicable." Mr. Jawadi could not do so, because prior art such as Schneider-US expressly recognized the applicability of Long to file-based systems, as discussed

above. As reflected further in other prior art, a POSITA understood the differences between file-based and sector-based approaches to backup, and also knew how to successfully apply concepts from one approach to another without being discouraged. *E.g.*, EX1025, 1:13-5:21 (describing conventional backup techniques for file-based and sector-based approaches as of 1995), 5:31-6:15 (describing application of conventionally file-based techniques such as incremental backup and individual file retrieval to sector-based backup).

23. Moreover, Kenley and Long incorporate aspects of both conceptual levels. For example, Kenley teaches a file-based backup–archiving system, and describes recording data allocations in disk sectors (EX1014, 16:5-32), and a file system that allocates storage blocks (sectors) for a file and updates a block map accordingly, where a block map links a file to block numbers (EX1014, 19:3-25). Contrary to Mr. Jawadi’s insistence that sector- and file-based solutions are incompatible (*e.g.*, EX2020, ¶63), Kenley expressly teaches that its system “also performs image recovery, which reconstructs an exact image of a directory (or file system or disk) as of the time of a particular past backup.” EX1014, 11:20-12:14 (describing how to use a file-based system to perform image recovery). Image backup and image recovery are typically considered sector-based approaches (*see, e.g.*, EX2020, ¶¶57-59 discussing this approach), but here Kenley expressly discloses using this concept from sector-based backup in a file-based system.

24. Similarly, Long’s preferred embodiment for backup–archiving is allegedly sector-based, but Long also describes a “modification request,” i.e., “a request from an application to modify the contents of the hard disk,” which spans from the supposed higher conceptual level (an application) to the lower conceptual level (the hard disk). EX1004, 7:28-29.

25. In general, a POSITA knew that it was common for methods described using one conceptual level to be applicable to other conceptual levels. For example, “copy-on-write” updates of data can be implemented at the sector level, the file level, and a database-table level, among others. *E.g.*, EX1003, ¶142 (quoting EX1007, 254). Further, regarding image or sector-based backup, it was well known that “[i]t is possible to combine the concept of image backup with file system backup.” EX1007, 256-257.

26. Despite Mr. Jawadi’s conclusory assertions of incompatibility, a POSITA would have been comfortable applying Long’s teachings to Kenley’s file-based system, and that is reflected in the evidence that I have discussed.

**4. Long Does Not Teach Away From
Combination With File-Based Systems**

27. I explained that “Long teaches a straightforward modification-driven archival approach that applied to known multi-tier backup–archival systems.” EX1003, ¶56 (citing EX1004, 11:54-56). Mr. Jawadi nevertheless argues that Long teaches away from a combination of Long with Kenley, based on the premise that

concepts from file-based systems are incompatible with sector-based systems, and vice versa. *E.g.*, EX2020, ¶¶138-141. As I discussed above, this premise is incorrect. *Supra* ¶¶12-16.

28. Mr. Jawadi cites no disclosure and could not do so, because no disclosure in Long criticizes, discredits, or otherwise discourages the combination of Long’s technique of triggering backup upon a modification request with a file-based backup–archival system such as Kenley.

29. Mr. Jawadi appears to refer to Long’s embodiment of “processing circuitry” (EX2020, ¶139), but this does not teach away from a combination of Long’s teachings with Kenley, because the disclosure of this particular embodiment would not discourage a POSITA from applying Long’s teachings on triggering archive operations with modification requests, which as discussed above come from higher-level applications in Long. Elsewhere, Mr. Jawadi alleges Long does not “disclos[e] detecting an instruction to perform an operation on an operating file,” and characterizes Long as “teach[ing] detecting access requests to alter data on storage device locations (sectors) but not files.” EX2020, ¶137 (citing EX1004, Abstract, 1:67-2:2, 10:16-19, 11:20-24). Here, Mr. Jawadi ignores Long’s teachings regarding a modification request, and consequently mischaracterizes Long’s teachings. Similarly, Mr. Jawadi also argues that “Long teaches operating at the lowest level, namely hardware level,” and focuses on Long’s disclosure of

“hardware-level fault tolerance and hardware-based error detection methods.” EX2020, ¶128 (citing EX1004, 3:3-18). Nothing about Long’s discussion of detecting *errors* would have discouraged a POSITA from implementing Long’s broader teachings regarding detecting *modification requests* “from an application to modify the contents of the hard disk” (EX1004, 7:28-29) in a file-based backup–archival system such as Kenley. Indeed, Long teaches copying “[t]he data to be modified on the disk ... one sector at a time” (EX1004, 7:37-38), but these are “operations performed by the audit subsystem in response to a modification request.” EX1004, 7:30-32. Long’s description of a modification request triggering a backup–archival process does not criticize, discredit, or discourage its use for triggering file backup–archiving in a file-based system. Neither does Long’s discussion of “error detection and masking” or “error containment.” EX1004, 2:64-68. On the contrary, Long describes these error detection and containment concepts as “one or more” elements of a general “fault tolerant strategy,” and lists them in parallel to recovery and repair. EX1004, 2:64-68; *id.*, 3:1-33 (describing error detection and masking and error containment as aspects of fault-tolerant systems distinct from error recovery).

5. Combining Kenley And Long Does Not Destroy The Principle Of Operation Of Kenley But Instead Improves It

30. Mr. Jawadi argues 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.” EX2020, ¶149. Mr. Jawadi’s arguments relate to “modifying Long to make it file based.” EX2020, ¶157. As mentioned above, this course is not the proposed combination.

31. Mr. Jawadi bases his arguments on the allegedly “well-known and fundamental differences between the file layer and sector layer,” and the unsupported assumption 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.” EX2020, ¶150. These assumptions, for which Mr. Jawadi cites no support, are incorrect because a POSITA knew that techniques and concepts from one conceptual level could easily be implemented in another, as I explained *supra* Section VI.A.3. Further, contrary to Mr. Jawadi’s insistence that “Long’s invention is deeply and intrinsically sector based” (EX2020, ¶155), this is a mischaracterization of Long, as nothing about the teachings in Long relied on in the Kenley-Long combination are fundamentally sector-based. Specifically, nothing about initiating a backup–archival procedure in response to a modification request, copying data into a buffer, or updating a database when such data is copied or migrated, is fundamentally tied to a sector-based implementation.

32. Although Mr. Jawadi alleges that “the proposed modifications would have changed the principle of operation of the modified reference,” Mr. Jawadi does

not present any specific arguments regarding the principle of operation of Kenley. EX2020, ¶155. As I explained in my First Declaration, the proposed combination implements Kenley’s file-based backup system with the modification-driven archival approach taught by Long, which achieves “increased reliability by creating backup data after each modification, thus avoiding the potential loss of data created in between the timed backups of Kenley.” EX1003, ¶¶49-51. The proposed combination does not destroy the principle of operation of Kenley in a way that renders the combination not obvious, but rather improves Kenley “by adding Long’s event-driven backup process that efficiently saved backups prior to any modifications of a file stored on hard disk.” EX1003, ¶52.

B. Kenley-Long Renders Element 1.4 Obvious

**1. Kenley-Long Searches For
The Archive File In Order To Move It**

33. Under the proper construction, Kenley-Long searches the first temporary storage location for the archive file because it must do so in order to move it from the memory buffer (first temporary storage location) to the audit buffer (intermediate storage location). As I explained in my First Declaration, “absent such searching, the system would neither know what it is moving nor when the move has completed.” EX1003, ¶66. Without searching for the archive file, the Kenley-Long system would not know whether there is anything to move to the audit buffer.

34. Even under Mr. Jawadi's improperly narrow construction, the Kenley-Long combination satisfies searching for the particular archive file. Long expressly teaches marking copied data with a marker header and marker trailer when storing it in the memory buffer. EX1004, 7:59-64; EX1003, ¶67. The marker header and marker trailer include metadata specifying that the copied data is audit data. EX1004, 5:30-67 (describing the marker header and the marker trailer), 6:46-48 (describing an "Audit" event). A POSITA would have implemented Kenley-Long such that a file is copied into the memory buffer with such marking metadata in order to mark (i.e., identify) the copied file (i.e., the archive file) as the specific archive file (e.g., as opposed to another version of the file). EX1003, ¶¶67-68 (discussing creation of the marker in Long and updating the audit header with the marker information). Further, a POSITA would have implemented Kenley-Long such that Kenley-Long updates a database to include the relevant marking metadata and the current location of the copied file (archive file) so that the copied file can be located, which facilitates "accurate retrieval and movement" of the file "for both backup and recovery." EX1003, ¶¶77, 55 (explaining how "a POSITA would naturally augment the catalog database of Kenley to store relevant information from the audit header taught by Long"), 58 (same), 68, 76-80 (explaining how Kenley-Long would update the database to indicate that the archive file is located in the memory buffer). Responsive to the triggering event (first event) to flush the memory buffer, Kenley-Long

searches the first temporary storage location for the particular archive file by (1) examining the database to identify what archive files are in the memory buffer, and (2) upon identifying presence of the particular archive file in the memory buffer (due to the database indicating that the particular archive file is located in the memory buffer), searching the memory buffer to locate the particular archive file (identifying the particular archive file based on its metadata) so that Kenley-Long may move the particular archive file from the memory buffer to disk. EX1003, ¶70; *see also* EX2020, ¶163 (Mr. Jawadi explaining how Long, in isolation, references the audit header to locate the copied data in the memory buffer). When searching the memory buffer for the particular archive file, Kenley-Long identifies the particular archive file in the memory buffer based on the particular archive file’s metadata marking it as such. Regardless of whether there is just one archive file or multiple archive files in the memory buffer waiting to be migrated to disk, Kenley-Long searches the memory buffer for each of them and migrates them accordingly. It must do so, in order to track the location of each archive file as it migrates through the different storage levels. *E.g.*, EX1003, ¶¶53, 76.

35. In addition to the use of marker information enabling the Kenley-Long combination to search the first temporary storage location for the archive file, and thus satisfying the claim element, “searching the first temporary storage location for the archive file responsive to the occurrence of a first event” is further obvious in

view of Long’s teachings regarding “having the memory buffer semaphored.” EX1004, 8:19-20; EX1003, ¶69. While Mr. Jawadi argues that “the presence of a semaphore would merely indicate that data is present” (EX2020, ¶168), which acknowledges that the use of a semaphore indicates presence of the archive file in the memory buffer, Mr. Jawadi misses the point about the semaphore. I explained that “a process checks that semaphore before writing to physical disk[,]which would constitute a ‘first event’” as recited in element 1.4. EX1003, ¶69. In this way, the memory buffer being semaphored (due to the archive file being present in the memory buffer) during the first event triggers Kenley-Long to search for the archive file in the memory buffer, so that it may migrate the archive file and resolve the semaphore. Mr. Jawadi is also incorrect that “[t]he semaphore would merely block the ability to write data *to the memory buffer* until the prior data in the memory buffer is flushed” (EX2020, ¶168)—the semaphore on the memory buffer blocks writes *to the hard disk*. See EX1004, 8:18-23.

36. Long specifically explains that “the memory buffer that *contains the audit markers*,” which contain the archive data, is flushed to disk. EX1004, 8:22-23 (emphasis added). The audit markers, rather than every bit of data in the entire memory buffer irrespective of whether it is an audit marker, in the memory buffer are flushed to the audit buffer on disk. The audit header in the memory buffer remains in the memory buffer, and is updated “to reflect the changes.” EX1004,

7:51-53. A POSITA would have understood that Long teaches selectively flushing the contents of the memory buffer, which requires searching for the particular data to be moved. Further, Long describes the use of a semaphore in the context of the audit method, rather than a general context, and a POSITA would have understood that Long specifically teaches “having the memory buffer semaphored” when there is an audit marker in the memory buffer. EX1004, 8:18-28 (describing the use of the buffer and the semaphore for properly capturing audit data), 7:27-8:17 (describing the “sequence of operations performed by the audit subsystem in response to a modification request”). A POSITA thus would have understood that Long teaches selectively flushing the audit markers from the memory buffer to disk.

37. Per Mr. Jawadi’s argument that the “[f]lushing” the memory buffer “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” (EX2020, ¶163), there is no basis for this conclusion. Mr. Jawadi does not explain how or why a POSITA implementing Long in Kenley’s file-based backup–archival system would blindly copy an archive file from the memory buffer to the audit buffer without first searching for the archive file that it is copying. Instead, Mr. Jawadi explains that “[t]he audit header tracks the beginning and the end of the Long memory buffer,” which “enables a block copy.” EX2020, ¶163; POR, 9-10. Long explains that “the audit subsystem writes the data stored in the memory buffer to the end of the audit

buffer 30 on disk,” where “the data stored in the memory buffer” refers to “[t]he data to be modified” that was copied sector-by-sector to “semiconductor memory buffer prior to writing the data to the audit partition on the hard disk.” EX1004, 7:37-53. A POSITA would have understood that “writing the data to the audit partition on the hard disk” would also occur sector-by-sector, rather than by a block copy. But even assuming *arguendo* that a POSITA would have used a block copy when implementing Long alone, identifying the portion of the memory buffer that is occupied by an archive file—based on the beginning and end of the file in the memory buffer—is equivalent to searching for the archive file in the memory buffer, because if the archive file was not occupying the portion of the memory buffer and thus not present in the memory buffer, there would be nothing to block copy. Furthermore, Mr. Jawadi’s arguments focus on implementing Long alone, and fail to consider the obvious implementation of Kenley-Long, where *files* are stored in the memory buffer. Typically, files are selected for migration from a storage location to a different storage location, and as such the system must search the storage location for the files in order to move them. *See, e.g.*, EX1014, 3:5-8 (disclosing “selecting files to be migrated”). Further, Kenley-Long updates a database to indicate the location of an archive file, and so would need to know, regardless of copy method, what files it moved from the memory buffer to the audit buffer on disk in order to appropriately update the database.

38. Regarding the memory buffer (temporary storage location), Long describes first storing data in “a semiconductor memory buffer” in order “[t]o increase the throughput of the audit subsystem,” because storing data in such a memory buffer is faster than storing directly to hard disk. EX1004, 7:43-46, 7:54-57. This “memory buffer,” which may be a “volatile memory buffer,” may also be used by the processing unit 14 “for another purpose.” EX1004, 7:47-49, 7:54-68. From this disclosure, a POSITA would have understood that the semiconductor memory buffer is random-access memory (RAM), and that the RAM that is shared because of its use “for another purpose.” EX1004, 7:48-49; EX1027, 253 (noting that “in common usage, [memory] refers only to the fast semiconductor storage (RAM) directly connected to the processor”); EX2010, 262, 264 (describing “buffer memory” and noting “[t]his memory may also serve as variable storage or even code execution space for the CPU”). This aspect is complementary to Kenley, whose “primary storage element” is RAM. EX1003, ¶54 (noting the complementary tiers of storage in Kenley and Long, including “RAM or semiconductor memory buffer” as primary storage); EX1014, 2:44-50 (describing the primary storage element of a hierarchical storage configuration), 13:48-50 (“RAM 30”), 14:60-61 (noting “RAM buffer cache” as “block 30 of FIG. 2”), FIG. 2.

39. For all the reasons described in my First Declaration and elaborated upon above, a POSITA would have implemented Kenley-Long to search the

memory buffer (first temporary storage location) for the archive file responsive to a first event, in order to move the archive file to the audit buffer on the secondary disk storage, where the archive file is stored. EX1003, ¶70.

**C. Kenley-Long Renders
Elements 1.6, 1.9, And 1.10 Obvious**

40. Elements 1.6, 1.9, and 1.10 relate to “updating a database to indicate that the archive file” is located in the first temporary storage location, the intermediate storage location, and the second storage location, respectively. My First Declaration explained how a POSITA would have found it obvious, in view of the complementary teachings of Kenley and Long, to update a database with location information for an archive file in order to enhance “transparent access” to the archive file as it migrates between storage elements. EX1003, ¶¶54-55, 76-80, 89-95. Mr. Jawadi’s arguments against Kenley-Long miss the forest for the trees by narrowly focusing on select language in the individual references and misinterpreting the proposed combination by assuming bodily incorporation of the references.

**1. Kenley-Long Updates A Database To
Indicate The Location Of The Archive
File In Order To Track The Archive File**

41. As my First Declaration explained, Kenley teaches a database storing catalogs of backup files that “enhances transparent access to file attribute data, independent of whether the file is stored in the secondary storage element or has been previously migrated from the secondary storage element.” EX1003, ¶55

(quoting EX1014, 3:25-28). After creating backup copies of files, Kenley updates the database with “a catalog of all files backed up,” and these database backup catalogs are used in Kenley to locate backup files for retrieval. EX1014, 8:26-29, 10:43-52 (cited by EX1003, ¶55).

42. Mr. Jawadi argues that Kenley’s catalogs would simply “indicate which files are stored in a particular group but do not disclose the location of any particular archive file.” EX2020, ¶173. But “location” as defined and used by the ’478 patent broadly refers to a general area on a storage device where a file is present, rather than a particular position of data within a storage device. *E.g.*, EX1001, 4:30-31 (defining “Storage Location” as “any storage device, or a buffer, folder, directory or designated area on a storage device”), 6:29-32 (describing how updating a database “to indicate the location of the archive files” means “to indicate that the archive files are resident in input buffer 20”), 6:36-38 (describing updating the data “to indicate that the archive files are now stored in the output buffer”). The backup catalogs described by Kenley store the location of files in the same way by indicating which files have been “copied to specific backup savesets,” thus allowing individual files to be located and retrieved. EX1014, 6:35-40 (describing the database–backup catalogs), 10:34-61 (describing using the backup catalogs in the database to locate and retrieve archive files). Further, the “file attribute data” stored and maintained by these catalogs includes a “backup location identifier” comprising various location

information (e.g., “the volume ID, the file starting offset, storage length and the file length”) that “enables the backup processor 12 to properly backup and restore files.” EX1014, 3:17-28, 7:45-62, 8:8-13, 8:25-28; EX1003, ¶53. And a POSITA would have understood that such a catalog would store location information enabling the system to locate the file. EX1027, 67 (defining “catalog” to include “location of files”). Without indicating the location of archive files, Kenley’s database would be useless.

43. Similarly, Long teaches updating an “audit header with ‘information that is used to manage the audit buffer 30, archive media ... and the general environment of the audit subsystem.’” EX1003, ¶46 (quoting EX1004, 4:42-47). Long teaches updating the audit header to indicate the location of copied data in the memory buffer (temporary storage location), audit buffer on disk (intermediate storage location), or archive media (second/permanent storage location). See EX1003, ¶¶76-77, 89-93. These updates in Long “facilitate[] the audit subsystem’s accurate retrieval and movement of data for both backup and recovery.” EX1003, ¶77.

44. Mr. Jawadi repeatedly admits that the audit header is updated with location information for data when it is placed into the memory buffer, in urging that “the audit header tracks the beginning and the end of the Long memory buffer” (EX2020, ¶163), “the Audit Header stores the information on the first and the last

markers” (EX2020, ¶164), “continues to track the beginning and the end of the buffer” (*id.*), and “information regarding the last marker ... will be used to update the Audit Header” (*id.*). *See also* EX2020, ¶¶174-175, 177, 179, 184. Regardless of Mr. Jawadi’s narrow fixation on the specific audit header fields for the first marker and the last marker in Long, Mr. Jawadi clearly agrees that Long teaches updating the audit header every time that a marker comprising data (archive file) is created and placed in the memory buffer (temporary storage location), moved from the memory buffer to the audit buffer (intermediate storage location), and moved to archive media (second/permanent storage location). EX1003, ¶¶76-77, 89-93.

45. As explained in my First Declaration, when implemented for a file-based system such as Kenley’s, the updates-upon-storing taught by Long in the database taught by Kenley would naturally indicate the current location of the archive file, thereby ensuring that the archive file can be reliably tracked and located. *E.g.*, EX1003, ¶¶76-80, 89-95. Consequently, the combined teachings of Kenley and Long would have suggested to a POSITA that elements 1.6, 1.9, and 1.10 were obvious.

VII. GROUND 2: CLAIMS 1-6 AND 8-11 ARE OBVIOUS IN VIEW OF CABRERA, SCHNEIDER, AND WHITING

46. Mr. Jawadi disputes the motivation and reasonable expectation of success of the proposed combination; of the challenged claims, Mr. Jawadi only disputes that Cabrera-Schneider-Whiting identifies presence of the archive file

(element 1.4) and transmits the archive file to a permanent storage location responsive to a second event (element 1.5). Mr. Jawadi otherwise does not dispute that Cabrera-Schneider-Whiting renders obvious the challenged claims.

**A. A POSITA Would Have Found The
Cabrera-Schneider-Whiting Combination Obvious**

**1. Cabrera-Schneider-Whiting
Improves Upon The Operation Of
Cabrera By Preserving Files Before Modification**

47. Cabrera teaches a file backup–archiving method that is initiated upon a “triggering event,” as well as various embodiments for such triggering event including “a command to stage data” for backup–archiving or generally “any other triggering event.” EX1005, 10:56-61. My First Declaration explained how a POSITA “would have been motivated to use a file-manipulation event as taught by Schneider as a triggering event to initiate the backup–archival process taught by Cabrera, such that the Cabrera backup–archive process takes place[] in response to a file modification, because doing so would ... ‘eliminate the need to pause and make backups or decide which files should be backed up.’” EX1003, ¶133 (quoting EX1020, 5:24-25). The resulting combination thus provides various advantages, including “increas[ing] the ease and reliability of backup and archiving.” EX1003, ¶¶133, 139-140.

48. Mr. Jawadi alleges that “Cabrera relates to periodic backing up of data” and modifying Cabrera as proposed is not obvious because it would “fundamentally

alter the operation of Cabrera.” EX2020, ¶¶221-224. But the teaching in Cabrera quoted by Mr. Jawadi as evidence of this alleged “periodic nature” itself indicates the use of periodic backup as an embodiment “[i]n the alternative.” EX2020, ¶221 (quoting EX1005, 3:45-50). Mr. Jawadi offers no explanation of how Cabrera is fundamentally periodic. Cabrera’s disclosure of using periodic backup as an *alternative* embodiment does not render it a fundamental aspect of Cabrera that would discourage and demotivate a POSITA from the proposed combination.

49. Mr. Jawadi also alleges that the proposed combination “would eliminate steps 124 and 126 from Cabrera,” which relate to determining whether a threshold amount of data is ready for backup. EX2020, ¶223; *see* EX1005, 10:39-49. But even Cabrera explains that “decision block 124 and time delay 126” are just one embodiment (“when a given amount of local storage is utilized” as the trigger), and that yet other mechanisms for triggering backup–archiving can be used. EX1005, 10:45-61. Further, these two decision blocks are a different embodiment than the alternative “periodic” embodiment taught by Cabrera. EX1005, 10:49-55.

50. Moreover, the principle of operation of Cabrera is better understood as the efficient backing up and archiving of files to remote storage. *E.g.*, EX1005, Abstract, 14:35-40. Even if the use of elapsed time were the only example triggering event for initiating backup–archiving disclosed by Cabrera (it is not), a POSITA still would have been motivated to improve Cabrera by using the known technique or

“fundamental method[] of recording the original state of information prior to its being altered” taught by Schneider, because doing so would predictably preserve files before modification and improve reliability of backup–archiving. EX1020, 34:12-13; EX1003, ¶¶137-143. Rather than waiting for time to elapse to initiate backup, the proposed combination ensures that the prior state of a file is preserved before the file is modified.

51. Finally, the proposed combination does not completely replace the other triggering events taught by Cabrera, which may still be in place in Cabrera-Schneider-Whiting to establish a baseline backup. For example, new data produced by data producers may be backed up or archived according to one or more of the triggers taught by Cabrera. My First Declaration does not say otherwise. Instead, Cabrera-Schneider-Whiting improves upon Cabrera alone by adding the file-manipulation event as a triggering event to ensure that a file on disk is copied before it is modified, thereby improving reliability “in the context of short term information recovery.” EX1003, ¶139 (quoting EX1020, 5:24-25).

**2. A POSITA Would Have Known That
Cabrera-Schneider-Whiting Would Work As Proposed**

52. A POSITA would have had a reasonable expectation of success implementing the file-modification event as taught by Schneider within the archive–backup system of Cabrera, because POSITAs knew how to intercept file operations and use them to trigger other actions. EX1003, ¶134 (citing EX1020, 30:4-11;

EX1021, 4:2-7, 20-25). Mr. Jawadi explained during his deposition that capturing file operations would have been easy for a POSITA to implement. EX1026, 236:20-239:2 (discussing Interrupt 21 routine and how to implement file modification intercepts to perform other routines); *see also* EX1021, 4:2-25 (describing same). Mr. Jawadi nevertheless argues that the “proposed combination would not work with the system of Cabrera” because “intercept[ing] a disk write before such data is written ... is fundamentally incompatible with the underlying operations of Cabrera.” EX2020, ¶225. This argument fails, as none of Mr. Jawadi’s observations about Cabrera explain how or why the proposed combination would not work.

53. Mr. Jawadi argues that the proposed combination would not work because “Cabrera only discloses moving data that has already been written by its data producers.” EX2020, ¶¶225-227. Mr. Jawadi argues that “Cabrera’s Data File has already been created / written before Cabrera’s back-up procedure is started” (EX2020, ¶228), and notes that the data file is “stored in a local storage medium, such as the hard disk for the local computer system.” EX2020, ¶228 (quoting EX1005, 6:63-64). Naturally, the file is already stored on the disk—otherwise, there would be no “file on the disk” to copy “before completing the write to the disk.” EX1003, ¶138. In view of Schneider’s teachings, when a file on the disk is about to be modified or overwritten through a write, the Cabrera-Schneider-Whiting combination creates an archive copy of the file “before completing the write to the

disk” and thus overwriting the file on disk. EX1003, ¶¶138-139, 141-143. Nothing about the Cabrera-Schneider-Whiting combination forbids other triggering events from initiating backup–archiving, so backup–archiving could still be initiated by “a command to stage data” or via other triggering events. Instead, Cabrera-Schneider-Whiting simply ensures that “historic versions of a file” are backed up or archived before they are modified. EX1003, ¶128.

54. Mr. Jawadi’s primary evidence for this argument is FIG. 2 of Cabrera, which depicts copying of a data file to the archive system. EX2020, ¶231. Mr. Jawadi points out that that, in Cabrera, the data file is copied to the archive system “after the downward facing arrows from the Data Producer to the Data File.” EX2020, ¶231. This position is uncontroversial, as Cabrera-Schneider-Whiting also copies the file from the disk to create the archive file, and so naturally the data file must already be on the disk. Further, Mr. Jawadi acknowledged that the arrows from the data producer to the data file also represent “editing of the Data Files in Cabrera” and thus may correspond to disk writes that would overwrite the data files already stored on disk. EX2020, ¶231. Mr. Jawadi’s observations about Cabrera’s FIG. 2 in isolation do not provide any basis for the Cabrera-Schneider-Whiting combination not working, or otherwise indicate that a POSITA would not have had a reasonable expectation of success implementing the combination.

**3. Whiting's Express Teachings Emphasize
Obviousness Of The "Database" Claim Elements**

55. Mr. Jawadi argues that "a POSITA would not combine Whiting with Schneider and Cabrera." EX2020, V.D. I disagree. My First Declaration explained that, although Cabrera does not expressly use the term "database," Cabrera discloses a "master file table" (MFT) storing location information for files. EX1003, ¶135. The database elements of the challenged claims are obvious in view of this disclosure alone. Whiting's express teachings regarding a database "provide implementation details for storing and updating the location of various archived versions of a file, including as the archive files are migrated from local storage to remote storage, so that the archive files can actually be located and accessed when needed by the system or user." EX1003, ¶135. As such, Whiting's teachings emphasize how obvious the database elements would have been to a POSITA, who knew that "the location of a backup–archive copy needed to be tracked in order to facilitate access to the backup–archive copy," so that the migration of archive files through a hierarchical storage system is "transparent to the system" as the archive files move through the system. EX1003, ¶135 (quoting EX1006, 6:52-60). Without updates to the database to track the location of archive files as they are moved to different storage areas, the database would not be able to facilitate transparent access to the archive files.

B. Cabrera-Schneider-Whiting Evaluates The Staging Area, And Thus Searches For The Archive File

56. Cabrera discloses evaluating the staging area to determine whether more data should be copied from the local storage area on hard disk. EX1003, ¶144; EX1005, 11:22-28. Cabrera performs this evaluation “based on any triggering event” (a first event). EX1005, 11:24-27. Further, the determination of whether to append more data depends on how much storage space is allocated or allocable for the staging area, which in turn depends on what files are already stored in the staging area. *E.g.*, EX1005, 2:12-13, 2:44-49, 2:55-61. Under the proper construction as discussed *supra* Section I.A, Cabrera’s evaluation of the staging area necessarily requires searching for an archive file previously stored in the staging area but not yet migrated; otherwise, the system would not know whether there is allocated or allocable space available to add the additional archive files to the staging area. EX1003, ¶144. This evaluation in Cabrera thus satisfies the element “searching the first temporary storage area for the archive file,” and the proposed Cabrera-Schneider-Whiting combination renders obvious element 1.4.

57. Mr. Jawadi notes that “staging area” and “staging file” discussed in Cabrera are different terms that refer to the same storage area. EX2020, ¶¶202-204. I agree. But regarding Mr. Jawadi’s argument that the staging file does not “correspond[] to any particular file from the local disk” (EX2020, ¶204), the proposed combination in my First Declaration does not identify the “staging file” as

the archive file. Rather, the copy of the data file stored in the staging area is the archive file. EX1003, ¶¶141-143.

C. Cabrera-Schneider-Whiting Moves The Archive File To A Permanent Storage Location Responsive To A Second Event, And Moves The Archive File To An Intermediate Storage Location Along The Way

58. The proposed Cabrera-Schneider-Whiting combination moves the archive file from the staging area to remote storage in response to a second event, which satisfies element 1.5. EX1003, ¶145. Mr. Jawadi argues that this movement of the archive file from the staging area to remote storage occurs without the archive file being stored in an intermediate subsystem, and that “[n]othing 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.” EX2020, ¶¶214-215. This argument is baseless, as Cabrera explains that transferring the data (the archive file) from the staging area to remote storage may be implemented by “delivering the data ... to an intermediate system or subsystem which will, in turn, deliver the appropriate data to remote storage.” EX1005, 7:52-60; EX1003, ¶¶124, 149.

59. Although Mr. Jawadi suggests that challenged claim 1 requires moving the archive file from the intermediate storage location to the remote storage location responsive to a second event, elements 1.5 and 1.8 do not specify such a requirement. Instead, element 1.5 simply recites “moving the archive file to a [permanent] storage

location responsive to a second event,” while element 1.8 simply recites “moving the archive file from the first temporary storage location to an intermediate storage location.” These elements are satisfied by the initiating transfer of the archive file from the staging area (temporary storage location) to the remote storage (permanent storage location) responsive to a second event, where the archive file is moved from the staging area to the intermediate system (intermediate storage location) as part of the transmission, as performed by Cabrera-Schneider-Whiting. *See* EX1005, 7:47-60, 11:46-55; EX1003, ¶¶145, 149.

I, David Maier, do hereby declare and state, that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code.

Dated: 24 October, 2024

A handwritten signature in cursive script, reading "David Maier", is positioned above a horizontal line.

David Maier