

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

DROPBOX, INC.,
Petitioner,

v.

ENTANGLED MEDIA, LLC
Patent Owner.

Case No. IPR2024-00285
U.S. Patent No. 8,484,260

PETITION FOR *INTER PARTES* REVIEW OF

U.S. PATENT NO. 8,484,260

CHALLENGING CLAIMS 1-8

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	MANDATORY NOTICES	1
	A. Real Party-in-Interest (37 C.F.R. §42.8(b)(1)):	1
	B. Related Matters (37 C.F.R. §42.8(b)(2)):	1
	C. Lead Counsel (37 C.F.R. 42.8(b)(3) & 42.10):	2
	D. Service (37 C.F.R. 42.8(b)(3)-(4)):	2
III.	SUMMARY OF THE '260 PATENT	2
	A. Brief Description of the '260 Patent	3
	B. Summary of the Prosecution History	4
IV.	THE PRIOR ART	5
	A. Havewala – Ex-1005	5
	B. Adams – Ex-1006	7
	C. Saridakis – Ex-1007	7
	D. Rothman– Ex-1008.....	8
V.	REQUIREMENTS FOR <i>INTER PARTES</i> REVIEW UNDER 37 C.F.R. §42.104.....	8
	A. Grounds for Standing 37 C.F.R. §42.104(a) and Fees.....	8
	B. Identification of Challenge under 37 C.F.R. §42.104(b) and Relief Requested.....	8
VI.	DISCRETIONARY DENIAL WOULD BE INAPPROPRIATE.....	9
	A. §325(d)	9
	B. §314(a).....	9
	C. <i>General Plastic</i> Does Not Apply	9
VII.	LEVEL OF ORDINARY SKILL	10
VIII.	CLAIM CONSTRUCTION	11
IX.	GROUND 1: HAVEWALA IN VIEW OF ADAMS RENDERS CLAIMS 1, 4-8 OBVIOUS.....	11
	A. Claim 1	11

1.	1[P]: “A process for operating on files located on multiple devices using a singular file system”	11
2.	1[A]: “accepting a request to operate on a file at a first device, wherein the file is selected from the singular file system on the first device”	20
3.	1[B]: “modifying the singular file system on the first device to make local files and virtual files appear indistinguishable to the singular file system, the local files and virtual files sharing a same location on the first device”	22
4.	1[C]: “intercepting the request by a software client on the first device”	32
5.	1[D]: “determining by the software client if the file is physically located on the first device or if the file is a virtual file of a corresponding file physically stored on a second device by reviewing file metadata”	34
6.	1[E]: “wherein a visual representation of the singular file system on the first device is identical to a visual representation of the singular file system on the second device”	40
7.	1[F]: “if the file is the virtual file of the corresponding file physically located on the second device, requesting by the software client on the first device that a peer-to-peer connection be brokered by a server-based web service between the first device and the second device”	43
8.	1[G]: “if the peer-to-peer connection is brokered, transferring the corresponding physical file from the second device to the first device”	48
9.	1[H]: “performing the operation on the transferred corresponding physical file at the first device.”	53
B.	Claim 4	56
C.	Claim 5	58
D.	Claim 6	60
E.	Motivation to Combine Havewala and Adams	61

F.	Claim 7	64
1.	7[P]: “A non-transitory computer-readable storage medium storing a set of instructions that, when executed by a processor, cause the processor to perform operations, comprising:”	64
2.	7[A]-[H]	65
G.	Claim 8	66
1.	8[P]: “A client comprising:”	66
2.	8[A]: “a memory;”	66
3.	8[B]: “at least one processor configured to:”	67
4.	8[C]-[J]	67
X.	GROUND 2: HAVEWALA IN VIEW OF ADAMS AND SARIDAKIS RENDERS CLAIMS 2 AND 3 OBVIOUS	68
A.	Claim 2	68
B.	Claim 3	70
C.	Motivation to Combine Havewala, Adams, and Saridakis	71
XI.	GROUND 3: HAVEWALA IN VIEW OF ADAMS, SARIDAKIS AND ROTHMAN RENDERS CLAIMS 1-8 OBVIOUS	73
A.	Claims 1-8	73
B.	Motivation to Combine Havewala, Adams, Saridakis, and Rothman	77
XII.	CONCLUSION	79

PETITIONER'S EXHIBIT LIST

Exhibit	Description
Ex-1001	U.S. Patent No. 8,484,260 (the “’260 patent”)
Ex-1002	File History of the ’260 patent
Ex-1003	Declaration of Dr. Patrick McDaniel
Ex-1004	CV of Dr. Patrick McDaniel
Ex-1005	U.S. Patent Publication 2007/0016621 (“Havewala”)
Ex-1006	U.S. Patent Publication 2002/0046232 (“Adams”)
Ex-1007	U.S. Patent No. 8,874,691 (“Saridakis”)
Ex-1008	U.S. Patent Publication 2005/0289218 (“Rothman”)
Ex-1009	Dropbox, Inc.’s <i>Sotera</i> Stipulation
Ex-1010	The Linux Information Project, “Metadata Definition.” 2006.
Ex-1011	Wayback printout, “NTFS Master File Table (MFT)”, NTFS.com
Ex-1012	Eberspächer, Jörg and Schollmeier, Rüdiger. “Peer-to-Peer Systems and Applications.” (2005), Chapter 5 First and Second Generation of Peer-to-Peer Systems.
Ex-1013	Frystyk, Henrik. “The World-Wide Web.” (1994).
Ex-1014	Wayback printout, “POSIX® 1003.1 Frequently Asked Questions (FAQ Version 1.10)”
Ex-1015	Sandberg, Russel. “The Sun Network File system: Design, Implementation and Experience.” (2001).
Ex-1016	Wayback printout – “File systems”
Ex-1017	Webpage printout – “Multics”

Ex-1018	Sandberg, Russel, et al. "Design and implementation of the Sun network file system." Proceedings of the summer 1985 USENIX conference. 1985.
Ex-1019	[Reserved]
Ex-1020	Wayback printout – Napster
Ex-1021	Shuler, Rus. "How Does the Internet Work?" (2002).
Ex-1022	Fox, Geoffrey and Pallickara, Shrideep. "Peer-to-Peer Interactions in Web Brokering Systems." (2002).
Ex-1023	Wayback printout – Introduction to Linux Loadable Kernel Modules
Ex-1024	Webpage printout – Deploying your hardware and software systems
Ex-1025	Wayback printout – Invasion of the Data Snatchers
Ex-1026	Warrick, R. Drake. "File Management: My Computer and Explorer." (2002).
Ex-1027	MacManus, Richard. "Microsoft has 97% of OS market, says OneStat.com." (2006)
Ex-1028	Alan Freedman, Computer Desktop Encyclopedia, Ninth Edition, Osborne/McGraw-Hill, 2001.

I. INTRODUCTION

Pursuant to 35 U.S.C. §311 and 37 C.F.R. §42.100 *et seq.*, Petitioner, Dropbox, Inc. (“Petitioner”), requests *Inter Partes* Review (“IPR”) of claims 1-8 of U.S. Patent No. 8,484,260 (“’260 patent,” Ex-1001) (“Challenged Claims”).¹

II. MANDATORY NOTICES

A. Real Party-in-Interest (37 C.F.R. §42.8(b)(1)):

The real party-in-interest is Dropbox, Inc.

B. Related Matters (37 C.F.R. §42.8(b)(2)):

Purported Patent Owner Entangled Media, LLC (“Patent Owner” or “PO”) has asserted the ’260 patent and U.S. Patent No. 8,296,338 (the “’338 patent”) against Petitioner in *Entangled Media, LLC v. Dropbox, Inc.*, Civ. No. 5-23-cv-03264-PCP (N.D. Cal., filed Dec. 16, 2022). Petitioner has also filed a Petition challenging the ’338 patent in IPR2024-00284. Third-party Unified Patents, LLC has also filed an unrelated request for *Ex Parte* Reexamination of the ’260 Patent (Application No. 90/015,221) on April 17, 2023 (the “Unified Reexam”).

As of the filing of this petition, no other judicial or administrative matters are known to Petitioner that would affect, or be affected by, a decision in an IPR of

¹ Because the ’260 patent’s priority is before March 16, 2013, the pre-AIA statute applies.

the '260 patent.

Entangled Media, LLC purports to own all rights, title and interest in the '260 patent, *see id.*, ECF No. 6-1, ¶7. Copies of this Petition and all supporting exhibits have been sent via Priority Mail Express to each identified party.

C. Lead Counsel (37 C.F.R. 42.8(b)(3) & 42.10):

Petitioner designates Dion M. Bregman (Reg. No. 45,645) as lead counsel, and designates Michael J. Lyons (Reg. No. 37,386) and Austin L. Zuck (Reg. No. 81,341) as backup counsel.

D. Service (37 C.F.R. 42.8(b)(3)-(4)):

Service Address: Morgan, Lewis & Bockius LLP, 1400 Page Mill Road, Palo Alto, CA (Telephone: 650.843.4000; Fax: 650.843.4001).

Pursuant to 37 C.F.R. 42.8(b)(4), Petitioner consents to e-mail service at: MLB-Dropbox-IPR-Team@morganlewis.com.

Pursuant to 37 C.F.R. 42.10(b), a Power of Attorney accompanies this Petition.

III. SUMMARY OF THE '260 PATENT

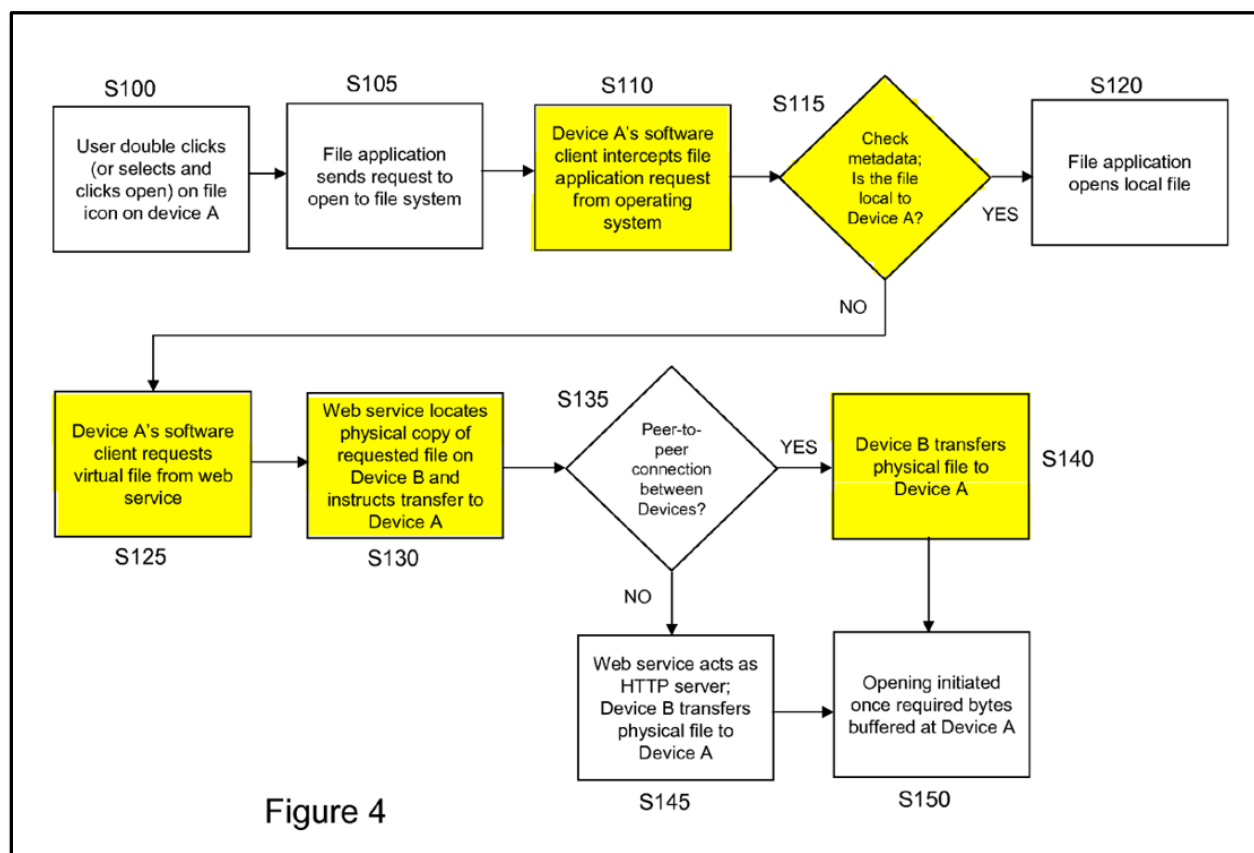
The application resulting in the '260 patent was filed on March 19, 2012, and is a divisional application of application No. 12/774,231, filed on May 5, 2010 which claims priority to provisional application No. 61/175,489, filed on May 5, 2009 (hereinafter, the purported "Critical Date").

A. Brief Description of the '260 Patent

The '260 patent claims a system and method for maintaining a file system that stores a physical file on one device and enables access to the physical file on other devices. The '260 patent states that data “synchronization across all devices for all content on all devices is limited or even prohibited by space restrictions.” Ex-1001, 1:57-59. The '260 patent proposes a system that “ensures that all [of] a user’s devices appear to share one single native file system containing all [of] the user’s files across all devices” while “not requir[ing] physical data replication across multiple devices.” *Id.*, 2:16-24. This was not new, however, as such systems were well known prior to the Critical Date.

The '260 patent’s claimed system is shown in Figure 4²:

² All emphasis and coloring added unless otherwise indicated.



The '260 patent illustrates in Figure 4 above that when a user selects a virtual file on Device A, a software client intercepts the request (S110) to determine whether the file is locally stored on Device A. If the file is not local, the software locates and requests the corresponding physical file from the device where it is locally stored, here Device B (S125, 130). The corresponding physical file is then transferred from Device B to Device A (S140). Ex-1003, ¶¶43-54.

B. Summary of the Prosecution History

During prosecution, the claims were rejected as being anticipated by U.S. Patent Publication No. 2006/0161585 to Clarke and also rejected as obvious by

Clarke in view of U.S. Patent Publication No. 2007/0050069 to Aboulhosn. After an Examiner interview, claim 1 was amended to include “modifying the singular file system on the first device to make local files and virtual files appear indistinguishable to the singular file system, the local files and virtual files sharing a same location on the first device.” Ex-1002, p. 50. The Applicant argued that the client in Clarke “distinguishes and discriminates locally stored object[s] and shared objects,” *id.*, p.54, that the file system in Aboulhosn “uses the file name extension ‘vf’ to discriminate and distinguish between virtual files and local files,” *id.*, p.55, and therefore, both Clarke and Aboulhosn were distinguishable from the amended claim 1.

To gain allowance, a subsequent Examiner’s amendment incorporated the limitation of a dependent claim requiring determination of whether a file is a virtual file “by reviewing file metadata” into the independent claims. Ex-1002, pp.18, 20; Ex-1003, ¶¶56-57.

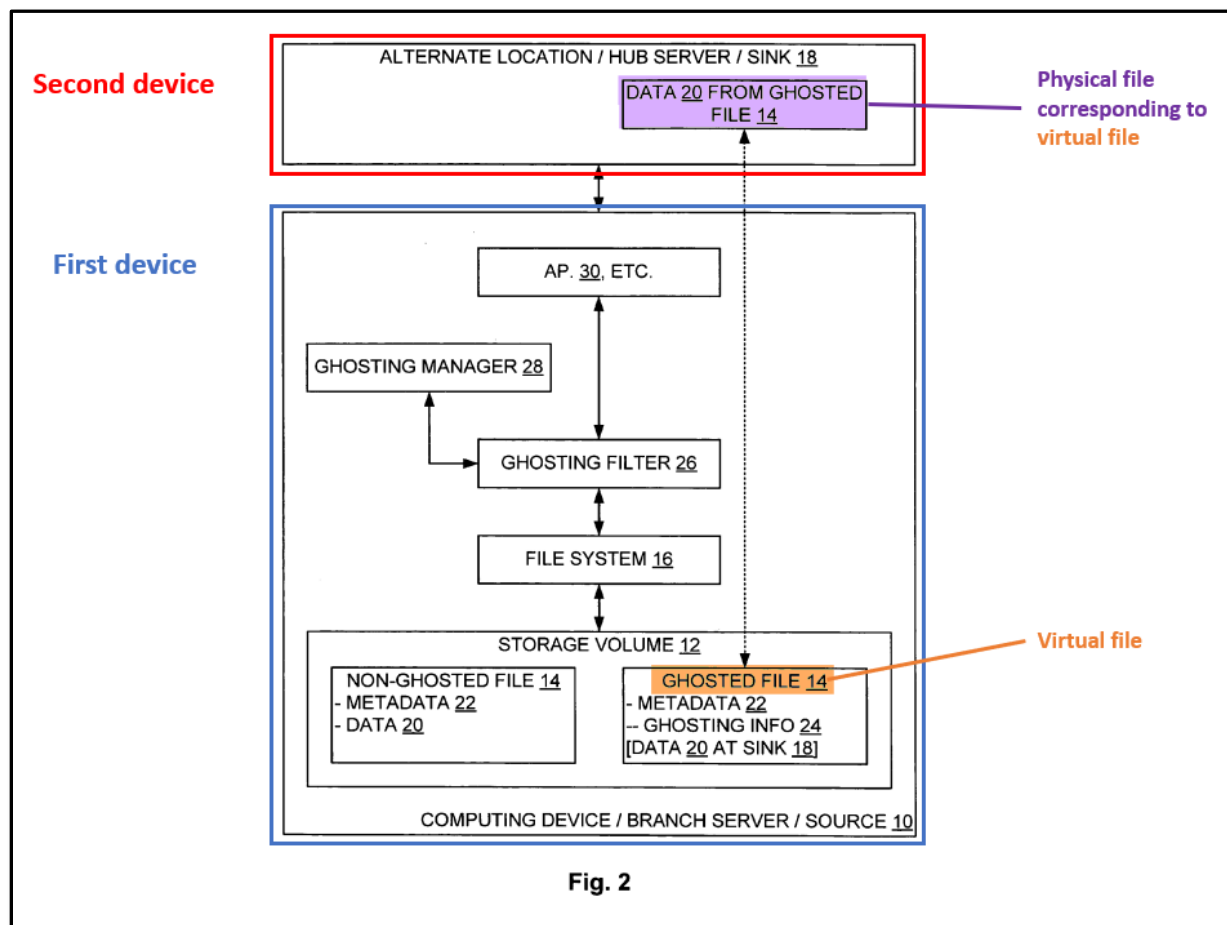
IV. THE PRIOR ART

A. Havewala – Ex-1005

U.S. Patent Publication No. 2007/0016621 (“**Havewala**”) published on January 18, 2007 and is prior art under at least 35 U.S.C. §102(b).

Havewala discloses a file system displaying both physical files located on a source (first device) as well as “ghosted” (*i.e.*, virtual) files that correspond

physical files stored at an alternate location or sink (second device). Ex-1005, FIG. 2. To maintain the appearance of the presence of files on a device, Havewala discloses “ghosting” a file by moving data from the “files to the alternate location or [] copy[ing] the entire file to the alternate location.” *Id.* ¶12; FIG. 2.



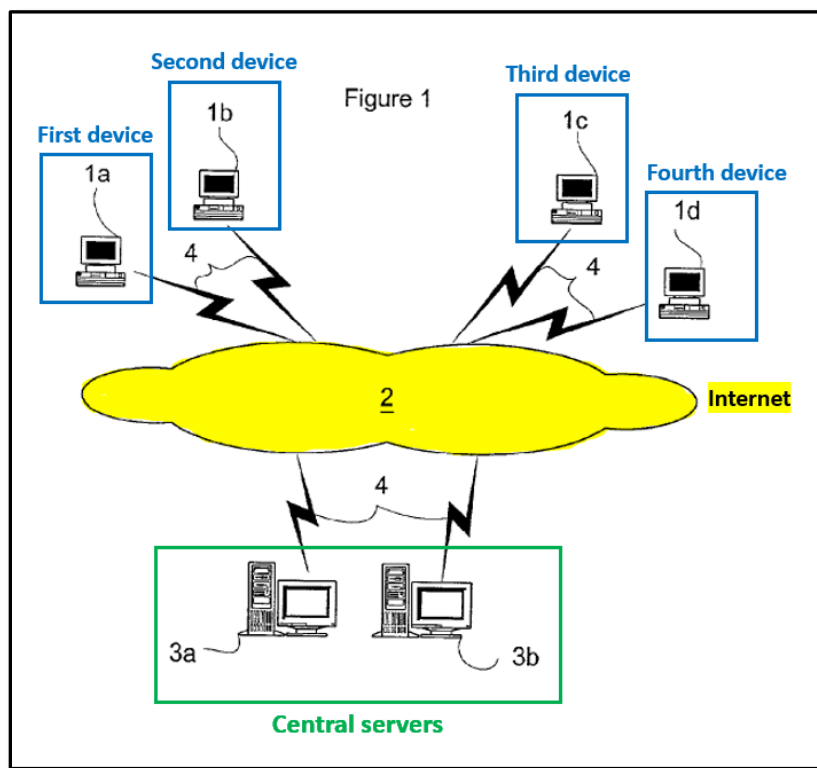
Ghosted and non-ghosted files are “organized by, accessed through, and otherwise controlled by a file system.” Ex-1005, ¶41. When an application requests to operate on the ghosted file (virtual file) at the source, a physical file is transferred from the “alternate location 18” to the source 10 and opened in an

application on the source. Ex-1005, ¶¶60; Ex-1003, ¶¶59-61, 22-52.

B. Adams – Ex-1006

U.S. Patent Publication No. 2002/0046232 (“**Adams**”), published April 18, 2002, is prior art under at least §102(b).

Adams discloses a system for “locating desired files in distributed file sharing over a computer network” (Ex-1006, Abstract) through a “peer-to-peer network connection” that utilizes a “central index server.” *Id.*, ¶25; Ex-1003, ¶63.



C. Saridakis – Ex-1007

U.S. Patent No. 8,874,691 (“**Saridakis**”), published October 28, 2014, is prior art under at least §102(b).

Saridakis discloses a method of “provid[ing] a peer-to-peer communication channel between peers utilizing hypertext transfer protocol (HTTP).” Ex-1007, Abstract. Saridakis discloses an alternative connection for “circumventing network obstacles,” such as firewalls, that may at times impair peer-to-peer communication channels. *Id.*, 2:48-53; 3:7-9; Ex-1003, ¶65.

D. Rothman– Ex-1008

U.S. Patent Publication No. 2005/0289218 (“**Rothman**”), published on December 29, 2005, is prior art under at least §102(b).

V. REQUIREMENTS FOR *INTER PARTES* REVIEW UNDER 37 C.F.R. §42.104

A. Grounds for Standing 37 C.F.R. §42.104(a) and Fees

Petitioner certifies that the ’260 patent is available for IPR and that the Petitioner is not barred or estopped from requesting this IPR.

Any additional fees for this IPR may be charged to Deposit Account No. 50-0310 (Order No. 105817-8002).

B. Identification of Challenge under 37 C.F.R. §42.104(b) and Relief Requested

In view of the prior art and evidence identified below, IPR of claims 1-8 of the ’260 patent should be granted because claims 1-8 are unpatentable and should be cancelled. 37 C.F.R. §42.104(b)(1)-(2).

Ground	Statute	Reference(s)	Claims
1	§103	Havewala, Adams	1, 4-8
2	§103	Havewala, Adams, Saridakis	2-3
3	§103	Havewala, Adams, Saridakis, Rothman	1-8

VI. DISCRETIONARY DENIAL WOULD BE INAPPROPRIATE

A. §325(d)

Discretionary denial under §325(d) is unwarranted because none of the references relied upon in the Grounds were cited during prosecution of the '260 patent, the related '338 patent, or in the Unified Reexam.

B. §314(a)

Discretionary denial is also inappropriate under §314(a).

Sotera Stipulation. Following the guidance that “the PTAB will not discretionarily deny institution in view of parallel district court litigation” if a *Sotera* stipulation is filed (Director Vidal’s June 21, 2022 Memorandum), Petitioner herewith provides such a *Sotera* stipulation (Ex-1009).

C. General Plastic Does Not Apply

In at least one decision the Board found that the *General Plastic* (“GP”) factors do not apply here, where the previous challenge to the '260 patent was in an *ex parte* reexamination proceeding. *See Samsung Elecs. Co., Ltd. v. Netlist, Inc.*, IPR2022-00615, Paper 20, 21-22 (PTAB Oct. 19, 2022).

Even if the Board finds that the GP factors apply, they weigh heavily against denial. The Unified Reexam challenged only four of the eight claims challenged in this Petition (GP Factors 1, 3, 6), which uses entirely different references from, and is not informed by, the Unified Reexam (GP Factors 1-5). This Petition is being filed less than a week after Petitioner served its invalidity contentions in district court (GP Factors 2-3). *See Code200, UAB v. Bright Data Ltd.*, IPR2022-00861, Paper 18, 5-6 (PTAB Aug. 23, 2022) (precedential). Petitioner was not involved with the Unified Reexam and this is the first petition that Petitioner has filed challenging any claim of the '260 patent (GP Factors 2, 6). *See also Mercedes-Benz v. Carucel Investments, LP*, IPR2019-01404, Paper 12, 12 (PTAB Jan. 22, 2020) (finding even membership in Unified Patents did not amount to a relationship that justifies discretionary denial). No reasons exist that would interfere with the Board's ability to issue a Final Written Decision within a year of institution (GP Factor 7).

VII. LEVEL OF ORDINARY SKILL

A person having ordinary skill in the art ("PHOSITA") at the time of the alleged invention of the '260 patent would have been a person holding at least a master's degree in computer science or electrical engineering, or a related degree, and with at least two years training or experience with networking and file systems. Ex-1003, ¶21.

VIII. CLAIM CONSTRUCTION

The Board need only construe terms “to the extent necessary to resolve the disputed issues.” *Apple Inc., v. SEVEN Networks, LLC*, IPR2020-00285, Paper 10, 8 (PTAB July 28, 2020) (citing *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017)).

For the purposes of this proceeding only, Petitioner does not propose that any terms require construction, because the claims of the '260 Patent are unpatentable under any reasonable construction, and there are no constructions necessary to resolve any disputes.

IX. GROUND 1: HAVEWALA IN VIEW OF ADAMS RENDERS CLAIMS 1, 4-8 OBVIOUS

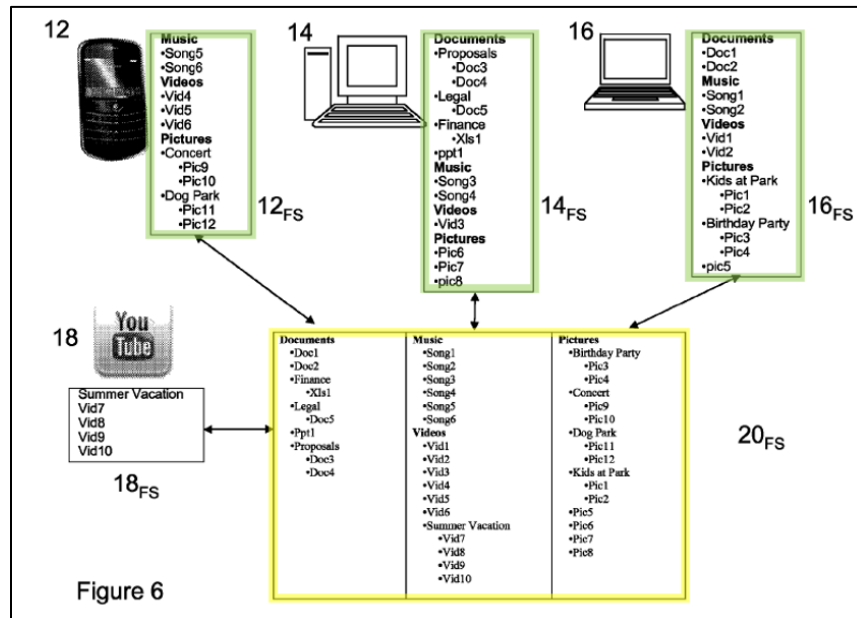
A. Claim 1

1. 1[P]: “A process for operating on files located on multiple devices using a singular file system”

To the extent the preamble is limiting, Havewala discloses it.

Singular File System. The '260 patent describes “a singular file system” as a system used to manage and display files across multiple devices, which is consistent with the plain and ordinary meaning of the terms “singular” and “file system.” *See, e.g.*, Ex-1001, 3:22-24, 5:67-6:1; 6:18-20. The '260 patent describes “file system” as “the component within an operating system responsible for managing all manner of files, as well as presenting them within a graphical user

interface.” *Id.* 4:30-32. As shown in Figure 6 below, the ’260 patent illustrates a singular file system using an “exemplary **unified file structure 20_{UFS}** [shown as 20_{FS} in yellow] created from the individual device file structures 12_{UFS}, 14_{UFS} and 16_{UFS} [green] and on-line service file structure 18_{UFS}.” *Id.*, 9:15-22.



Ex-1001, Fig. 6; Ex-1003, ¶86.

Havewala discloses the same. Havewala discloses implementing a system for remotely accessing files in “a computing device [that] has a storage volume, a **file system** managing the storage volume, and a file stored on the storage volume by the file system and accessed through such file system.” Ex-1005, ¶18; *see id.* ¶41 (“computer files 14, and the files 14 on the volume 12 are organized by, accessed through, and otherwise controlled by a file system 16 running on the computing device 10.”), ¶9. Havewala further explains that “a user or the like can

browse **the entire volume 12 of the source 10 by way of the file system 16** even if some of the files 14 on the volume 12 are ghosted and some of the files 14 on the volume 12 are not ghosted.” *Id.*, ¶51. The file system may “organize[] the volume 12 according to a **directory format** or otherwise” (*id.*, ¶50) and may display a file using “a graphic representation such as a ‘thumbnail’” (*id.* ¶52). Havewala also discloses that its system may be “practiced **in distributed computing environments** where tasks are performed by **remote processing devices** that are linked through a communications network.” Ex-1005, ¶34; Ex-1003, ¶87.

In particular, Havewala’s system contemplates a scenario where a first device—source 10—may include a ghosted (or virtual) file that “is present [on the source 10] in a reduced or ‘stub’ form” (*id.* ¶49) such that “source 10 contains all metadata 22 from the original, non-ghosted file 14, and also contains ghosting information 24, that may be employed to retrieve the data 20 for the file 14 from the sink 18” (*id.* ¶50). When a file is ghosted, the data with the metadata is transferred to a second device—alternate location/sink 18. *See, e.g., id.* ¶54 (when removing “primary data” from a non-ghosted file at the source and transferring the data with the metadata to the sink, “the entirety of such file 14 is stored at the sink 18”); *see also id.* ¶12. Ex-1003, ¶88.

Havewala uses different terminology throughout the specification to refer to both the “source 10” and the “alternate location 18.” For example, Havewala

refers to “source 10” as “computing device 10,” “storage device 10,” (Ex-1005, ¶43) and “branch server 10” (*id.*, ¶46). Havewala refers to “alternate location 18” as a “sink 18” or “hub server 18.” *Id.*, ¶48. Havewala further explains that “alternate location 18 may be local to the computing device or may be **remote therefrom**” and “may be another volume 12 on the storage device 10, **another volume 12 on another computing device 10**, a file warehouse at a server, a long-term storage device at a remote server, or the like.” *Id.*, ¶43. Therefore, as discussed in this Petition, “source 10” (and its associated uses of this term) refers to a first device (e.g., computing device) including the storage volume storing the ghosted file in its common file system; and the “alternate location 18” (or its associated uses of this term) refers to a second computing device including the storage volume storing the non-ghosted file in its common file system. Ex-1003, ¶89.

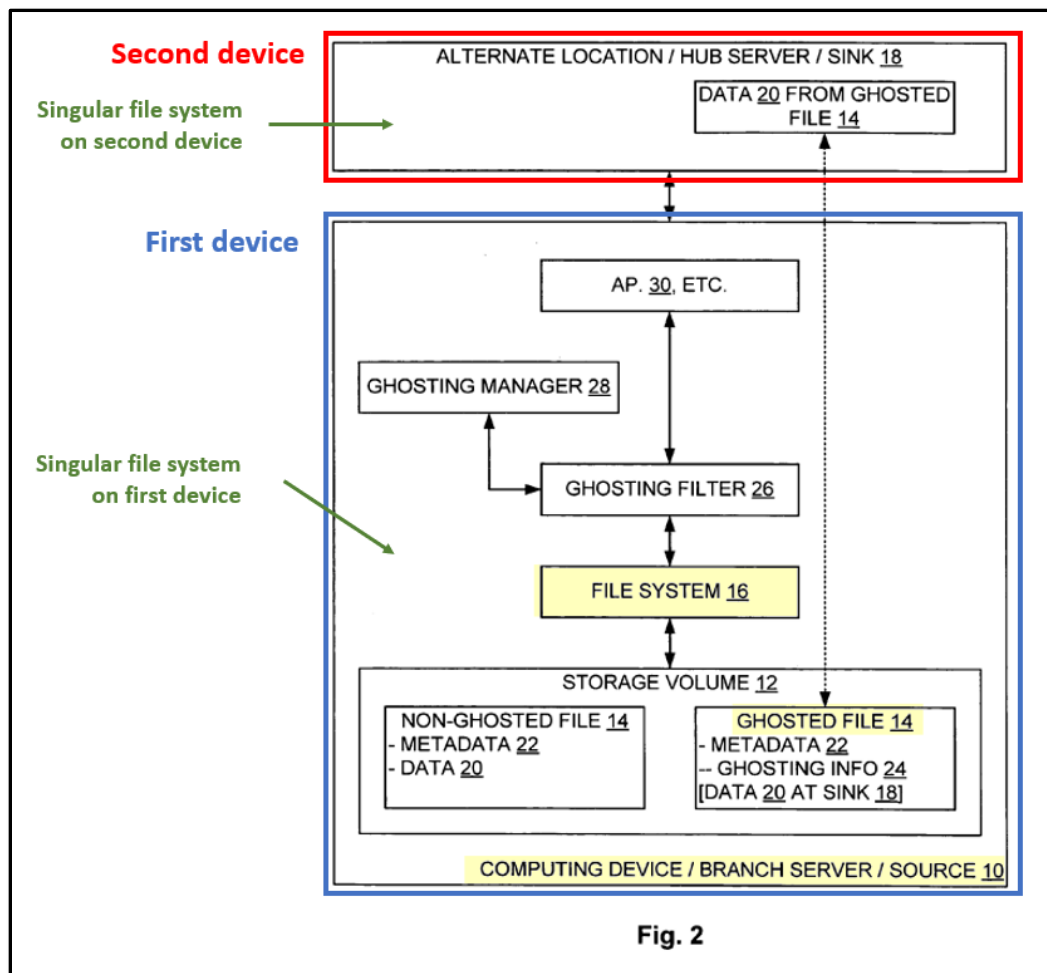
Havewala’s Disclosure of Implementing its Singular File System in Multiple Computing Devices. Havewala discloses that the alternate location may be “another volume 12 on **another computing device 10**” (Ex-1005, ¶43) and that “a **computing device** has a storage volume [and] a **file system** managing the storage volume” (*id.* ¶18). In other words, Havewala discloses that the source and sink may both be computing devices 10, that look and behave exactly the same, i.e., they have the same file system stored on both devices. For example, Havewala’s

system of ghosting a file is enabled through the installation of software programs and program modules onto a computing device with a pre-existing operating system (e.g., Microsoft Windows computer). *See* Ex-1005, ¶34; Ex-1003, ¶90.

Havewala explains that its system is designed to overcome problems in the prior art. Havewala explains that the prior art's replication of copies of files at multiple branch locations increases the "amount of traffic over the network...perhaps to a point that exceeds available bandwidth" and also that, "as the overall size of all the files at the hub server increases, it may in fact become the case that each branch server does not have enough space available thereon to store all of such files." *Id.* ¶14; Ex-1003, ¶91.

To overcome this problem, Havewala's system can be installed on multiple computing devices at those branch locations such that the computing devices would "be able to 'ghost' the non-relevant files at the branch server [e.g., first computing device/source 10] of the particular branch, such that each non-relevant file remains on the branch server [e.g., first computing device/source 10] in a smaller, 'ghosted' form." Ex-1005, ¶16; Ex-1003, ¶92. The "data [of the ghosted file on branch server/first computing device] is stored at a sink such as an alternate location or a hub server [i.e., second computing device]." Ex-1005, ¶17. If "the ghosted file is needed at the branch server [e.g., first computing device/source 10], the data for such ghosted file may be retrieved from the hub server [i.e., second

computing device/sink 18], the ghosted file may be reconstituted” and made available at the branch server/first computing device. *Id.*, ¶16. This is shown, for example, in annotated Figure 2 below. Ex-1003, ¶92.



Ex-1005, Fig.2.

Havewala explains that the system is designed to be utilized in, for example, situations where multiple different computers throughout a particular organization wish to have access to a file, but the organization wishes to reduce at least networking and storage constraints and store a file at only one computing device.

Ex-1003, ¶93. In such a situation, Havewala's programs and program modules are stored on similar devices (computing device 10 acting as a source, a sink, or both), for example computing devices running Windows operating software. *See, e.g.*, Ex-1005, ¶¶9, 34-40, p.1 (Assignee: Microsoft Corporation). In such a situation, the same file system (e.g., the Window's file system as modified by Havewala) is installed and operates on the multiple devices to enable ghosting files between devices as discussed herein in more detail. *Id.*; Ex-1026. Havewala confirms that just like the source, files can be modified on both the first device and the second device. Ex-1005, ¶119 ("the file 14 has been **modified at the sink**"), ¶127 ("if the data 20 of the file 14 at the source 10 is not modified but the corresponding data 20 at the sink 18 is modified"), ¶103.

Therefore, consider a scenario where a file is ghosted on Computer A (e.g., first computing device/branch server 10) on which Havewala's system has been installed and the physical data of the file is stored on Computer B (e.g., second computing device/alternate location/branch hub 18) also with Havewala's system installed. A request to open the non-ghosted file on Computer B using, e.g., Windows file system modified by Havewala, would result in the file being opened as usual because the physical contents of the file are present on Computer B. *See, id.*, ¶54 ("[W]here the hub server 18 replicates files 14 to the branch server 10, it may in fact be the case that while the primary data of the data 20 is removed from

the remainder of the replicated file 14 at the source 18 to form the ghosted file 14, the entirety of such file 14 is stored at the sink 18.”), ¶52, Figs.3A-3B.

Conversely, an attempt to open the ghosted file at Computer A (again using, e.g., Windows file system modified by Havewala) results in the physical file and/or associated data from Computer B being transferred to Computer A so that the file can be “reconstituted” and opened on Computer A. *See, e.g., id.*, ¶¶49, 24; Ex-1003, ¶94.

As discussed further below, Havewala explains that its system is installed such that any operation performed on a file (e.g., opening) is performed transparently, such that the computing device’s user need not know that Havewala’s system is being run or utilized when opening any particular file. *See id.*, ¶64 (“[O]perations may be performed **transparently** or with notice **to the user** or application 30, although **transparency is likely preferable to such user** or application 30.”). As a result, Havewala discloses a singular file system at each computing device that utilizes the same file system interface (e.g., Windows file system modified by Havewala) presenting a “directory” of files available across multiple devices in an organization in such a way that a user does not know whether the file is ghosted or physically present on a particular computing device.

Operating on Files. Havewala discloses performing certain commands (*i.e.*, operations) on files, for example through “issuing an open command to the file

system 16 with regard to the file 14.” *Id.* ¶66; *see also id.*, ¶66-69 (further discussing “open” and “read” operations); Ex-1003, ¶96. Modifying a file always requires opening it.

Havewala Also Renders Obvious a Singular File System. Havewala discloses a singular file system across at least the computing devices 10 and 18. To the extent PO argues that Havewala does not disclose a singular file system, it would have been obvious to a PHOSITA to implement Havewala’s invention using a **singular file system** across multiple devices. Ex-1003, ¶¶97-98. For example, an organization attempting to at least reduce overheads and storage use associated with file duplication and alleviate growing storage constraints would install Havewala’s system across its existing computing devices without altering the way that those devices operate. In addition to its other advantages, implementing Havewala’s system in this way would provide a seamless, integrated user experience that would minimize disruption of processing workflows and maximize data availability to all users of the system. *Id.*, ¶96.

Indeed, it would be preferable to implement the software—as Havewala discloses—“transparently” such that users and employees need not be burdened to search through various file storage locations or remember or be notified as to whether a particular file is physically present on their computing device (*i.e.*, if the file is a non-ghosted file) or if the file is actually stored remotely (*i.e.*, the file is a

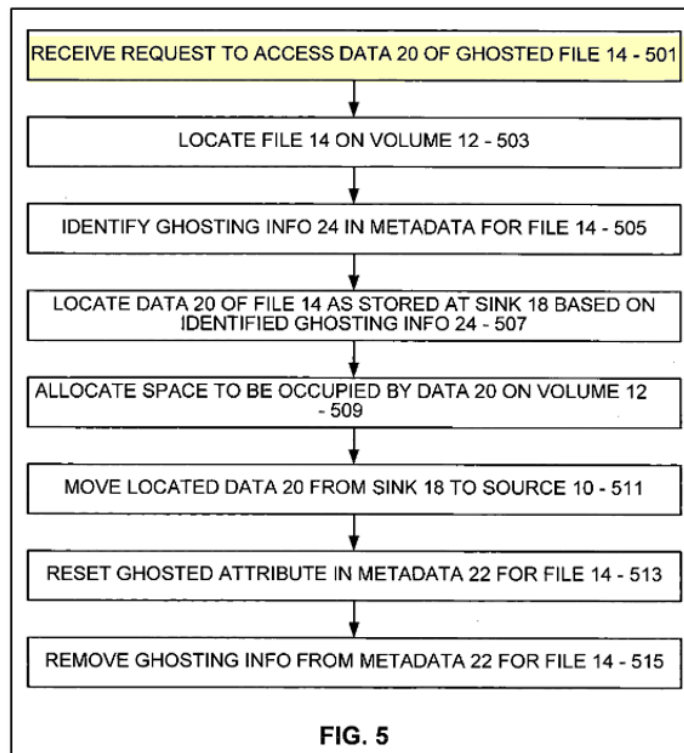
ghosted file). Ex-1003, ¶¶97-98. Thus, a PHOSITA implementing Havewala's system in exactly the type of situation that Havewala is designed for would have found it obvious to employ Havewala such that a user on any given computer in the network may, for example, browse for both non-ghosted and ghosted files in a single, coherent and common file system. *Id.*

Therefore, Havewala discloses a **“process for operating on files** [*e.g.*, Havewala's “open” commands] **located on multiple devices** [*e.g.*, first computing device 10 and second computing device 18] **using a singular file system** [*e.g.*, file system 16 present at the first and second computing devices 10 and 18].”

2. 1[A]: “accepting a request to operate on a file at a first device, wherein the file is selected from the singular file system on the first device”

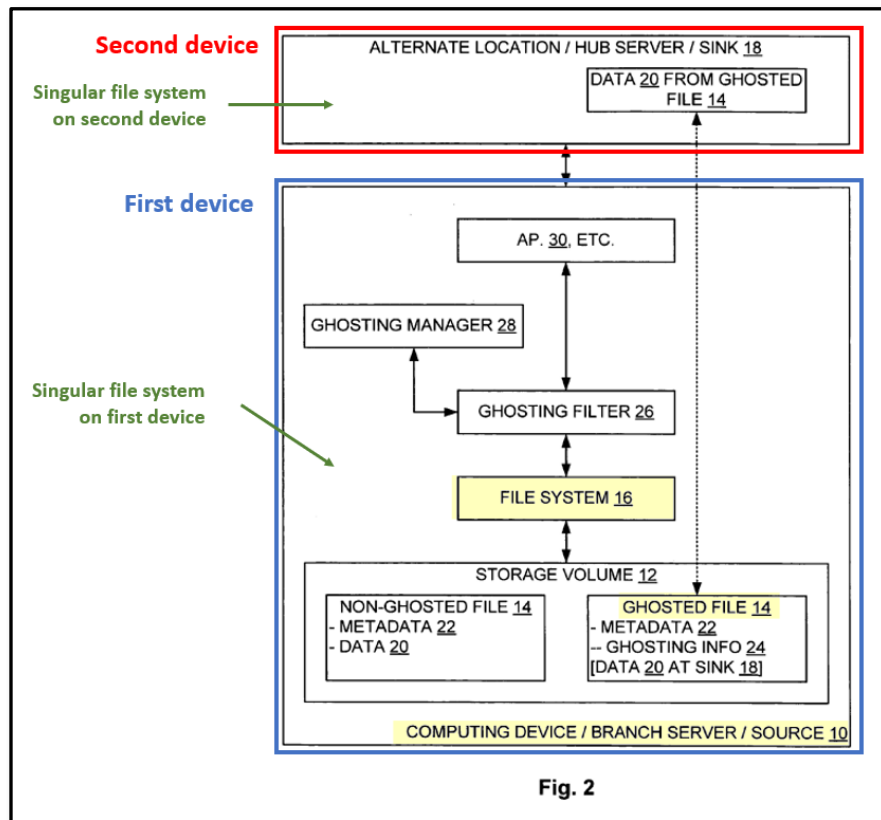
As detailed above, Havewala discloses a first device with ghosted and non-ghosted files and a singular file system on the first device to manage and perform operations on those files. Ex-1005, FIG 2. In one situation, ghosted (*i.e.*, virtual) files reside at the first computing device or source/branch server 10. *Id.*, ¶50 (“[G]hosted file 14 as **present on the volume of a source 10** is stored in the same location on the volume 12 as the original file 14.”). To access the file, a user selects a file from the file system and an “application 30 begins the process by **issuing the open command with respect to a particular ghosted file 14** on the volume 12 and such **open command ultimately is received by the file system**

16.” *Id.*, ¶¶67; *see also id.* FIG.2, FIG.5, ¶¶59, 60, 66. Figure 5, below, of Havewala confirms that its system “receive[s] [a] request to access” or open the file that is ghosted on the first device.



Ex-1005, FIG.5. Havewala therefore discloses that it accepts a request to open a file that has been selected from the singular file system on the first computing device or source/branch server 10. Ex-1003, ¶¶100-102.

As illustrated by Figure 2 below, Havewala’s singular file system on the first computing device (source/branch server 10) includes file system 16 and storage volume 12, which contains at least a ghosted file 14. *Id.*, ¶41.



Ex-1005, FIG.2.

Therefore, Havewala discloses “**accepting a request to operate on a file at a first device** [e.g., Havewala’s open command that is received and accepted by the first computing device or source/branch server 10], **wherein the file is selected from the singular file system** [e.g., ghosted file or non-ghosted file on file system 16] **on the first device** [e.g., computing device/source 10].”

3. 1[B]: “**modifying the singular file system on the first device to make local files and virtual files appear indistinguishable to the singular file system, the local files and virtual files sharing a same location on the first device**”

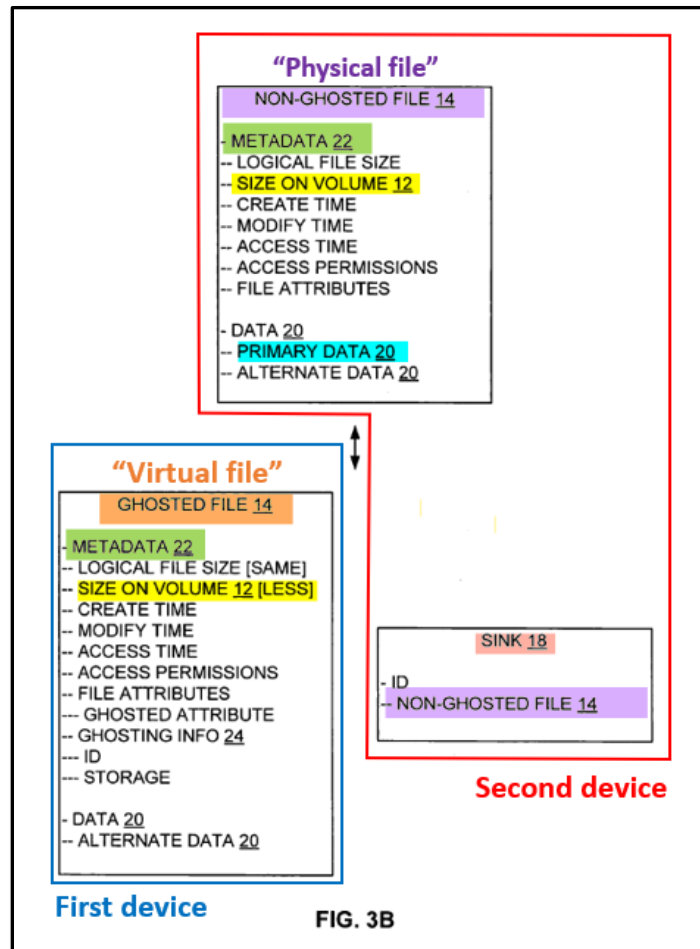
Havewala discloses this limitation.

Virtual and Local Files. The '260 Patent describes virtual files as files that “act as mock files and mock the property of the physical file.” Ex-1001, 8:5-6. These virtual files “appear to the user exactly as local files” and have “all that the physical file has except that it doesn’t take any space on the local device with the exception of the space required for meta-data storage.” *Id.*, 7:66-8:5.

Havewala’s “ghost files” correspond to the claimed “virtual files.” Havewala discloses that a local, non-ghosted file on a storage volume “include[s] data and metadata relating to the data.” Ex-1005, ¶18. To “ghost” a local file, “at least a portion of the data of the file is removed from the file and stored at an alternate location such that the data **does not occupy substantially any space on the volume** and the file is in a **reduced, ghosted format.**” *Id.*, Abstract; *see also id.* ¶¶12, 16, 18, 19, 44.

Further, as shown in Figure 3B, Havewala discloses that the **now-ghosted (i.e., virtual) file 14** has the same **metadata 22** (e.g., logical file size, create time, modify time, access time, access permissions, and file attributes) as the corresponding **non-ghosted (i.e., physical) file 14**, but that the **ghosted file** uses less storage space on the first device because it does not include, for example, the “**primary data 20**” of the physical, “**non-ghosted**” file that is stored on the second device. Ex-1005, FIG. 3B; ¶50 (“Significantly, **the ghosted file 14** as resident on the volume 12 of the source 10 contains all **metadata 22** from the original, **non-**

ghosted file 14.”).³



Ex-1005, FIG.3B.

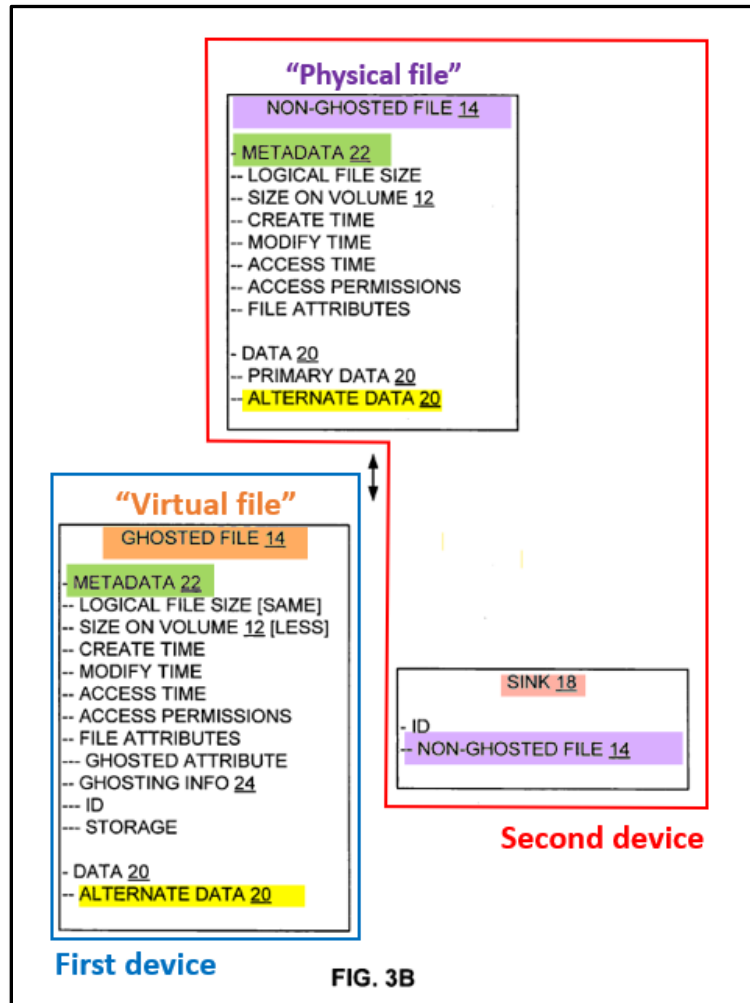
Indistinguishable to File System. The '260 patent states that “virtual files

³ As discussed below in Section IX(A)(5), Havewala discloses that “primary data” is transferred to and from the source (first device) and the sink (second device) to ghost and reconstitute a ghosted file. Primary data corresponds to the claimed physical file. Ex-1005, ¶¶52, 53; Ex-1003, ¶¶151-157.

appear in essentially all ways **the same** as a local file, making them **indistinguishable** unless such a distinction is enabled as a separate option.” Ex-1001, 6:38-40, 7:66. The ’260 patent states that a “virtual file includes a **thumb nail** with an icon...file name, the file size, the file type, the date it was created, the date it was modified, **all** that the physical file has except that it doesn’t take any space on the local device with the exception of the space required for **meta-data** storage” (*id.*, 7:66-8:5), including information concerning “whether the selected file is a local file or a virtual file” (*id.*, 8:35-38). Havewala discloses the same.

As shown in FIG. 3B, Havewala discloses that a ghosted file “contains **all metadata** 22 from the **original, non-ghosted file** 14.” Ex-1005, ¶50. Similar to the ’260 patent’s virtual file, the ghosted file includes the same metadata as the non-ghosted file such as “logical file size”, “create time”, “modify time”, and other “file attributes.” *Id.*, FIG. 3B. Further, the **ghosted file** on the first device (*i.e.*, **source/branch server 10**) retains the “**alternate data 20**” that is also possessed by the full **non-ghosted (physical) files** on the second device (*i.e.*, **alternate location/branch hub/sink 18**). Also like the ’260 patent’s metadata, Havewala’s “alternate data 20” retained on the first device “may include a graphic representation such as a ‘**thumbnail**’ that can be employed when display[ing] a representation of the file 14, perhaps by...the **file system**.” Havewala’s ghosted file’s metadata also includes “ghosting information” indicating that it is a ghosted

(i.e., virtual) file with a corresponding physical file which is located at an alternate location. *Id.*, ¶¶19, 20, 50.



The '260 patent describes that the file system indistinguishably accesses the virtual and physical files by “transparently perform[ing] numerous commands to request, locate and retrieve a physical copy of the virtual file without the knowledge of the requesting application or operating system.” Ex-1001, 8:60-64. To do so, the '260 patent states that “the software client intercepts the operation

request,” retrieves the remote file “from the remote device to the local device,” and performs “the operation...on the file at the local device.” *Id.*, 6:51-59.

Havewala discloses the same transparent operations. Havewala discloses that Havewala’s “operations may be performed **transparently**...to the user or application” and “**transparency** is likely preferable to such user or application 30.” Ex-1005, ¶64. “Notably, the application 30 in issuing such open command is **not expected to have any knowledge** that the particular file 14 **is in fact ghosted.**” *Id.*, ¶67. For example, the ghosted file’s “ghosting information...is not typically used by the [] inquiring application 30 or the like” (*id.*, ¶58), and therefore ghosted files appear indistinguishable from physical files to the file system.

In making a call to access a ghosted file, the operating system and its file system does not know that the ghosted file is any different from a local file, and, instead, it is the ghosting filter that intercepts the command and locates and reconstitutes the ghosted file:

when a user or application 30 tries to access the removed data 20 of a ghosted file 14, the file system 16 upon receiving a request for such access will discover that the ghosted file 14 does not contain the removed data 20 and thus return an error which the **ghosting filter 26 will intercept**. Upon such interception, then, **the ghosting filter 26 employs the file system 16 to obtain the ghosting information 24**

from the metadata 22 for the ghosted file 14, and based on the obtained ghosting information 24 such ghosting filter 26 triggers reconstituting of such ghosted file 14, as will be set forth in more detail below, so that the request for access may ultimately be honored.

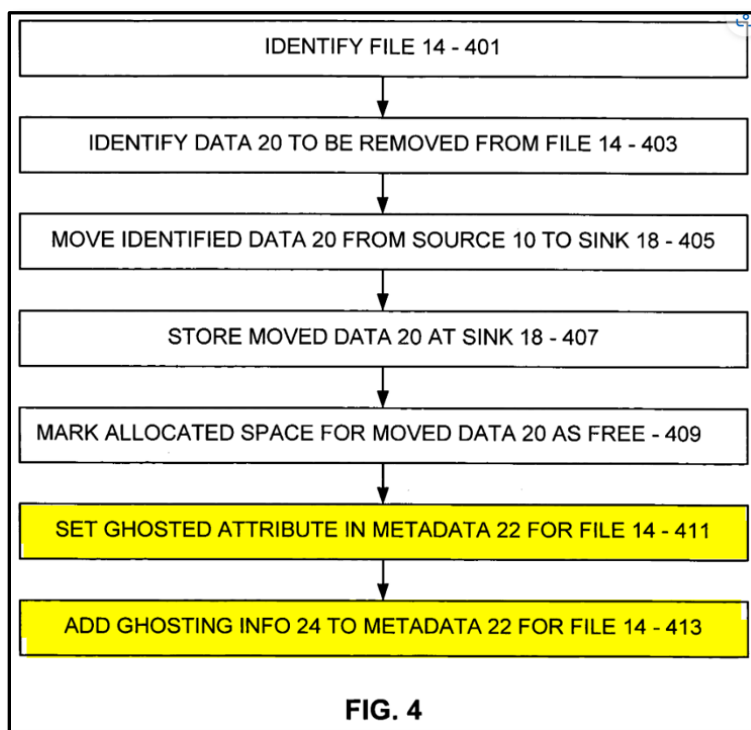
Ex-1005, ¶64.

To the extent PO argues that Havewala does not disclose virtual and physical files indistinguishable to a file system, a PHOSITA would have understood that Havewala's ghosted and non-ghosted files—which are indistinguishable to the user and application—are necessarily and inherently also indistinguishable to the file system. Specifically, the same commands to open files, read data, write data, and close files are used for both ghosted and non-ghosted files. *Id.* ¶66, FIG. 6, FIG. 7. Those operations, for example a request to open the file, are processed by the file system in the same way regardless of whether the file is local or virtual. *See* Ex-1005, ¶68. If the file is virtual, the file system operates in its typical fashion and—before the file system's error command is recognized by the file system, operating system, application, or otherwise—Havewala's ghosting manager and filter have intercepted the error, retrieved the ghosted file's data from the alternate location 18, and proceeded to open the reconstituted (or now-un-ghosted) file on the first device. *See id.* To the extent that PO argues that this is not inherent, a PHOSITA would have at least found it

obvious for the reasons discussed herein. Ex-1003, ¶112.

Modifying The Singular File System To Make Local Files And Virtual Files Appear Indistinguishable To The Singular File System. The '260 patent states that a “software client” “modifies the **natural behavior** of the operating systems of the individual registered devices in a way that, as it pertains to remote files, **prevents the operating system from doing what it normally does.**” Ex-1001, 7:43-50.

Havewala discloses modifying a file system in the exact same way. Havewala discloses that its system may be implemented by “computer-executable instructions, such a program module[], being executed by a computer.” Ex-1005, ¶34. Such program modules “include routines, programs, objects, components, data structures and the like that **perform particular tasks** or implement particular abstract data types.” *Id.* Havewala further discloses a “ghosting manager” and “ghosting filter” that are responsible for performing particular tasks related to, among other things, ghosting and reconstituting files. *Id.*, ¶¶64, 65. The ghosting manager “perform[s] the bulk of the **ghosting functionality** for the source 10 as is shown in connection with FIG. 4.” Ex-1005, ¶65. In particular, as shown in Figure 4, the ghosting manager amends “the metadata 22 for the now-ghosted file 14...to show that such file 14 is now ghosted (step 411)” (*id.*, ¶57), for example to include “ghosting information 24 (step 413).” *Id.*, ¶58.



Ex-1005, FIG.4.

Moreover, as discussed above, the operating system and file system are unaware if a file is ghosted because the ghosting filter intercepts an error when a command on the file is issued. The ghosting filter “aid[s]” the operations “that are performed at the source 10 on behalf of a user or application and with respect to a ghosted file.” *Id.*, ¶64. “Such operations may be performed **transparently** or with notice to the **user or application 30**, although **transparency** is likely preferable.” *Id.*; *see also id.*, ¶66. In performing these operations, the ghosting filter “receives [the] ghosting information” implemented by the ghosting manager to “create[] a buffer and a handle corresponding to the ghosted file 14, and deliver[] the handle to application 30 as the **(normal) response to the open command therefrom.**”).

Id., ¶68. Therefore, the ghosting manager and ghosting filter modify the file system such that local and ghosted files appear indistinguishable to the operating system, and subsequently to its file system. Ex-1003, ¶104-117.

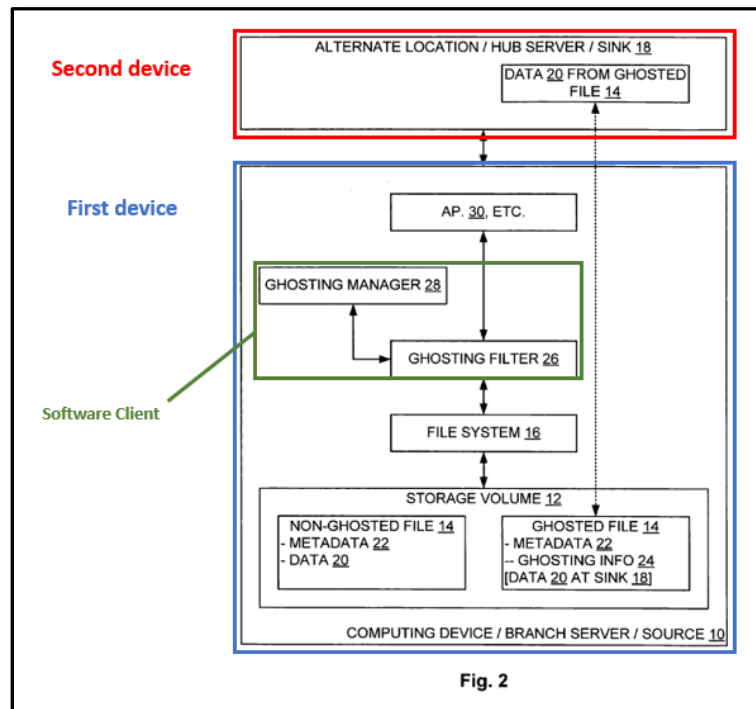
Same location. Havewala’s ghosted (virtual) files and non-ghosted (local) files share the same location at the source 10 (first device). The ’260 patent describes a “location” as a “directory/folder” within a “native file system.” Ex-1001, 1:64-2:1. Havewala discloses that “a **ghosted file 14** as present on the volume of a **source 10** is stored in the **same location** on the volume 12 as the **original file 14.**” Ex-1005, ¶50. “Accordingly, a user or the like can browse the **entire volume 12 of the source 10** by way of the file system 16 even if some of the files 14 on the volume 12 **are ghosted** and some of the files 14 on the volume 12 **are not ghosted.**” *Id.*, ¶51. Moreover, Havewala’s file system “organizes the volume 12 according to a **directory** format.” *Id.*, ¶50.

Therefore, Havewala discloses “**modifying the singular file system on the first device to make local files and virtual files appear indistinguishable to the singular file system** [*e.g.*, Havewala’s file system is modified so that local file and the ghosted file have metadata that makes them indistinguishable to the file system], **the local files and virtual files sharing a same location on the first device** [*e.g.*, ghosted and non-ghosted files on the volume 12 of source 10].

4. 1[C]: “intercepting the request by a software client on the first device”

The '260 patent states that a “Software Client (or the Client)—is a piece of software that is installed and runs continuously on the device.” Ex-1001, 4:46-47; *see id.*, 5:47-50. The '260 patent also states that the software client “modifies the natural behavior of the operating systems of the individual registered devices.”

Havewala’s Software Client. As discussed in Section IX(A)(3), Havewala discloses “program modules” that “perform particular tasks” in Havewala’s system, including those performed by a ghosting filter and ghosting manager. Ex-1005, ¶¶34, 64, 65, 140; *see id.* ¶34 (“[T]he invention is described in the general context of computer-executable instructions, such as program modules, being executed by a computer,...includ[ing] routines, programs, objects, components, data structures and the like that perform particular tasks or implement particular abstract data types.”). Havewala discloses that the source 10 (*i.e.*, the first device) includes the “ghosting filter” that aids in ghosting files and the “ghosting manager” that “includes additional ghosting functionality and direct access to network resources such as the sink 18.” Ex-1005, ¶¶64, 65. This is also shown in Figure 2, below. A PHOSITA would have recognized that Havewala’s ghosting filter and ghosting manager are “programs” and correspond to the claimed software client. Ex-1003, ¶120.



Ex-1005, FIG.2.

Intercepting A Request To Operate on a File. Havewala explains that:

when a user or application 30 tries to access the removed data 20 of a ghosted file 14, the file system 16 upon receiving a **request for such access** will discover that the ghosted file 14 does not contain the removed data 20 and thus return an error which the **ghosting filter 26 will intercept**. Upon such **interception**, then, the **ghosting filter 26 employs the file system 16 to obtain the ghosting information 24 from the metadata 22 for the ghosted file 14, and based on the obtained ghosting information 24 such ghosting filter 26 triggers reconstituting of such ghosted file 14 ... so that the**

request for access may ultimately be honored.”

Ex-1005, ¶64. Havewala further discloses that “the ghosting filter stores the read command in a queue or the like for awaiting further processing (step 613).” *Id.*, ¶71. Havewala’s ghosting filter—which is part of the software client on the first device—therefore intercepts a request to operate on a file through standard file system behaviors when a user requests access to a ghosted file that is not physically present on the first device. Ex-1003, ¶¶119-123.

Therefore, Havewala discloses “**intercepting the request** [*e.g.*, ghosting filter 26 intercepting an access request] **by a software client** [*e.g.*, Havewala’s ghosting manager and ghosting filter] **on the first device** [*e.g.*, computing device/source 10].”

5. **1[D]: “determining by the software client if the file is physically located on the first device or if the file is a virtual file of a corresponding file physically stored on a second device by reviewing file metadata”**

Havewala Discloses Virtual Files on a First Device and Physical Files on a Second Device. The ’260 patent describes a “virtual file” a file that “doesn’t take any space on the local device with the exception of the space required for meta-data storage (less than 1 kilobyte per file).” *Id.*, 7:66-8:6.

As discussed in Section IX(A)(3), Havewala discloses “a ghosted file 14 [*i.e.*, virtual file] as present on the volume of a source 10” (*i.e.*, the first device).

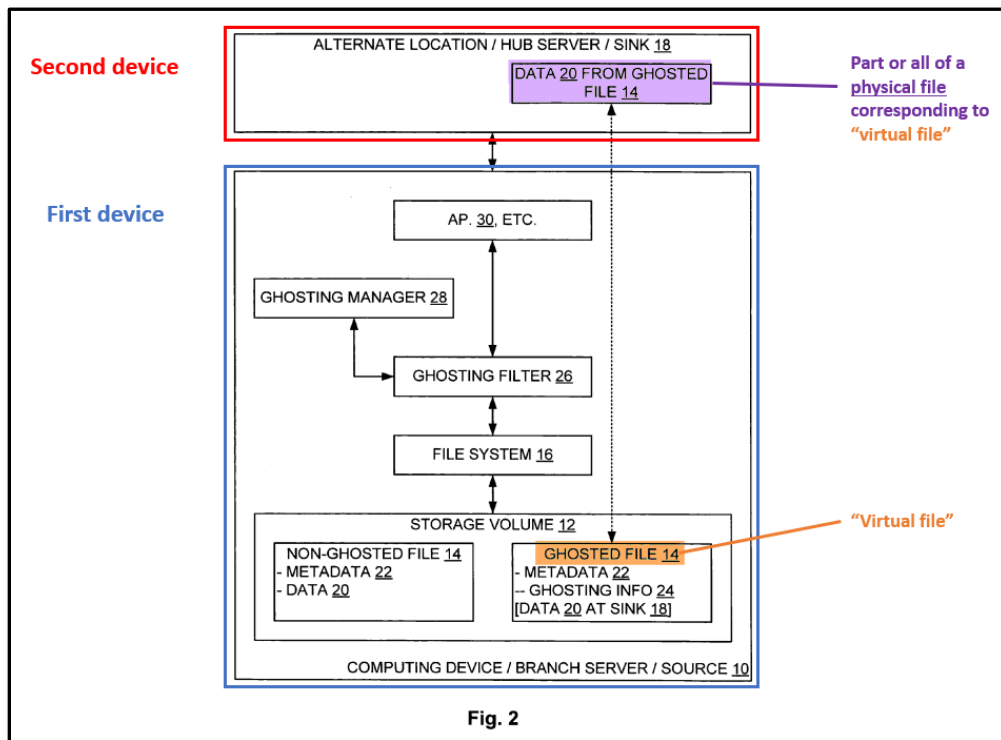
Ex-1005, ¶50. Havewala’s ghosted file, like the ’260 patent’s virtual file, has had its “data [] moved from the file to the alternate location for being stored thereat such that the file is in a reduced, ghosted form.” Ex-1005, ¶¶19-20. As a result, “[s]pace on the volume formerly occupied by the moved data is thus marked as being free.” *Id.* The data remaining on the source 10 (*i.e.*, the first device) is “alternate data,” such as the “metadata of the now-ghosted file,” including basic file information, “a graphic representation [of the ghosted file] such as a thumbnail” (*id.* ¶52), and ghosting information (*id.*, ¶¶19-20).

The ’260 patent describes a physical file as a “physically stored **data file**[]” that comprises the physical “content” of a virtual file “stored on a different device.” Ex-1001, 5:13-15, 2:7-9, 9:64-65, 8:32-33.

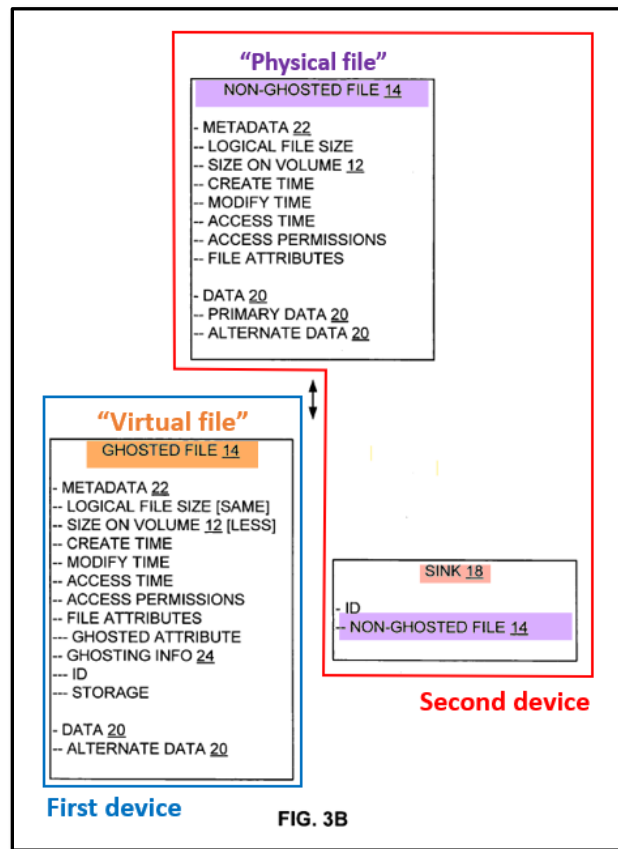
Havewala also discloses the same kind of physical file as the ’260 patent. When a non-ghosted file is ghosted (*i.e.*, local file becomes a virtual file), “the **primary data** of the data 20 is removed from the remainder of non-ghosted file 14 to form the ghosted file 14” and “such primary data is **stored at the sink 18.**” Ex-1005, ¶53. Removal of the “primary data” from a non-ghosted file to form a ghosted file at the source may also result in the “**entirety of the [non-ghosted] file [being] stored at the sink 18.**” *Id.*, ¶54. Moreover, “the primary data is the data 20 from the file 14 **as may be employed by an application**” to, for example, open the file. *Id.*, ¶52. Therefore, Havewala’s primary data is a physical file that is

physically stored on a second device (e.g., sink).

As shown in Figures 2 and 3B, Havewala further discloses that the “primary data” or “entirety” of ghosted file’s corresponding non-ghosted file (*i.e.*, the physical file) is stored at a remote alternate location (e.g., sink).



Ex-1005, FIG.2.

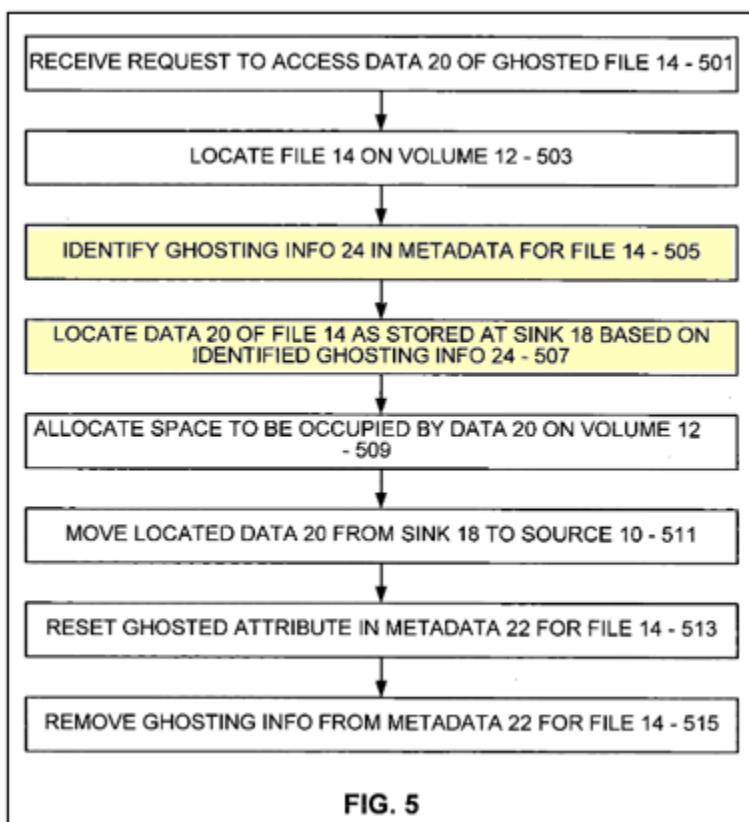


Ex-1005, FIG.3B.

Determining Location of Files via Metadata. Havewala discloses that once a file is ghosted (*i.e.*, a physical file is transferred to another device and a virtual file is left in its place), “the **metadata** of the now-ghosted file is **amended to include ghosting information** including information that may be employed to retrieve the **moved data for the file from the alternate location**” (*id.* ¶19) as well as information that “may include the ID stored with and identifying the **data 20** at the **sink 18**...[and] **how to locate the sink 18**” (*id.*, ¶58). *See also id.* ¶57, 58, FIGs. 3A-B. The ’260 patent similarly describes the claimed “virtual file” as

having metadata from which “the software client on Device A determines ... the remote device on which [the virtual file] is stored (Device B).” Ex-1001, 8:41-43.

As discussed in Section IX(A)(4), Havewala’s ghosting filter (a component of the software client) intercepts a request to operate on a ghosted file. Havewala further discloses that after such interception, the ghosting filter “obtain[s] the **ghosting information 24 from the metadata 22** for the ghosted file 14” (*id.*, ¶64) and that the “data 20 of the file 14 as stored at the sink 18 is **located** based on such identified ghosting information” (*id.*, ¶60). *See also id.*, ¶¶50, 51, 57. These operations are further shown in Figure 5, below, which are “performed with the aid of a **ghosting filter.**” *Id.*, ¶64.



Ex-1005, FIG.5.

Therefore, Havewala discloses “**determining by the software client** [e.g., ghosting filter 26] **if the file is physically located on the first device** [e.g., the non-ghosted file 14 on the source device is accessed by the user] **or if the file is a virtual file of a corresponding file physically stored on a second device by reviewing file metadata** [e.g., upon interception, the ghosting filter 26 determines from the ghosted file’s ghosting information 24 of the metadata 22 if the file is a ghosted file stored on the sink device].”

6. 1[E]: “wherein a visual representation of the singular file system on the first device is identical to a visual representation of the singular file system on the second device”

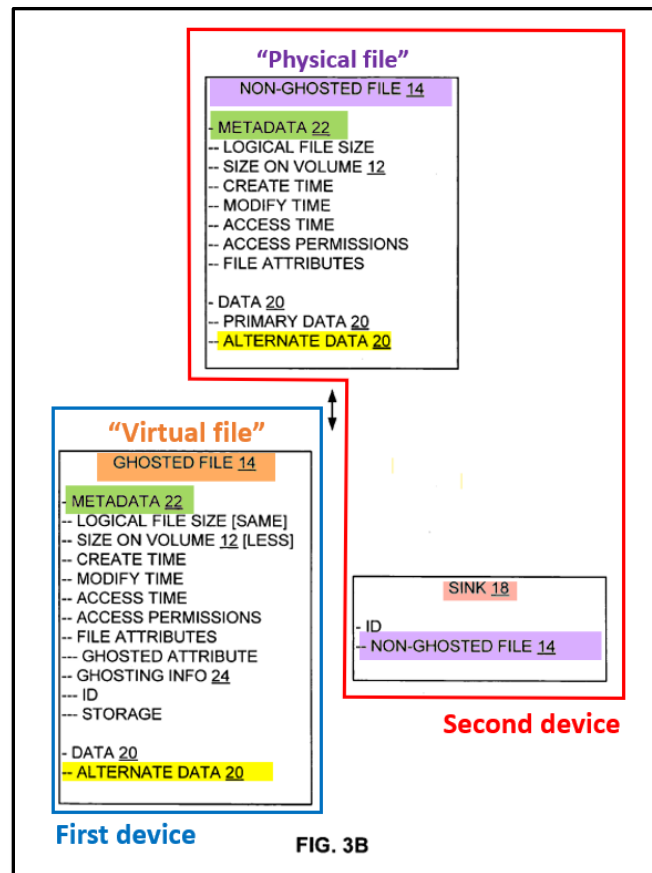
Havewala discloses this limitation. Havewala’s system is designed to overcome a prior art “**replication architecture**” where multiple branch servers 10 (i.e., a first computing device) are populated with “non-ghosted copies of all files 14 from a central hub server 18” (i.e., the second computing device). Ex-1005, ¶62. At least to save bandwidth, Havewala’s “**replication engine** may decide to simply **create ghosted copies of the files 14 on the branch servers 10**” (i.e., the first device). *Id.* As discussed above at length, incorporated here, Havewala’s system as integrated into an operating system utilizing, e.g., Microsoft’s file system, would present a user with the same file system interface in which the user would view the ghosted and non-ghosted files in the same way, and in the same directory/manner (e.g., in Microsoft’s file system), on the first computing device or branch server/source 10 as on the second computing device or alternate location/sink 18. *See* Section IX(A)(1) (1[pre]); Ex-1005, ¶¶12-17. Indeed, Havewala explicitly discloses that the ghosted and non-ghosted files are “stored in the **same location**” and that “a user or the like can browse the **entire volume 12 of the source 10** by way of the file system 16 even if some of the files 14 on the volume 12 **are ghosted** and some of the files 14 on the volume 12 **are not**

ghosted.” Ex-1005, ¶50-51.

Havewala further discloses that the ghosted file retains the metadata and alternate data (e.g., a thumbnail) of the corresponding non-ghosted file. *See Id.*, ¶51-53. “As but one example, **such alternate data** may include a **graphic representation such as a ‘thumbnail’ that can be employed when displaying a representation** of the [ghosted] file 14, perhaps by...the file system 16 **at the source 10.**” *Id.*, ¶52. Therefore, Havewala discloses presenting a thumbnail of a ghosted file at the first computing device (branch server/source 10) in the same way that a thumbnail of a non-ghosted file is presented. The same is true of the non-ghosted file on the second computing device or alternate location/sink 18. The user viewing files on the alternate location/sink 18 would find all ghosted and non-ghosted files in the same location and directory and would be able to open the file (which is local to the alternate location/sink 18) in the same way and with the same visual representation as it would be presented on the first device. *See id.*, ¶119 (“the file 14 has been modified at the sink”); Ex-1003, ¶¶136-137.

As shown by Figure 3B, below, the ghosted file maintains the same “alternate data” (e.g., including at least the graphic representation or thumbnail) as the corresponding non-ghosted file at the sink (second device). Ex-1005, ¶52. This confirms that a ghosted file and the file system at a branch 10 (first device) is visually represented in the same way as it would be at the hub (second device).

Thus, Havewala provides an identical visual representation of the singular file system across both devices such that not only are ghosted files on the first device in the same location(s) as non-ghosted files on the second device, but also ghosted files visually look the same as non-ghosted files. In other words, the visual representation of the file systems on both devices is identical at least because users would browse through the file system in the same way as if they were on the first or second device and both devices would show the same visual file information (i.e., icons, thumbnails, etc.).



Ex-1005, FIG.3B.

A PHOSITA would have understood that Havewala's replication engine creates identical visual representations of the singular file system across the first and second device at least because copies of ghosted files, containing the same alternate data (e.g., metadata) as its corresponding non-ghosted file, are replicated from the sink to the source. Ex-1003, ¶¶135-140.

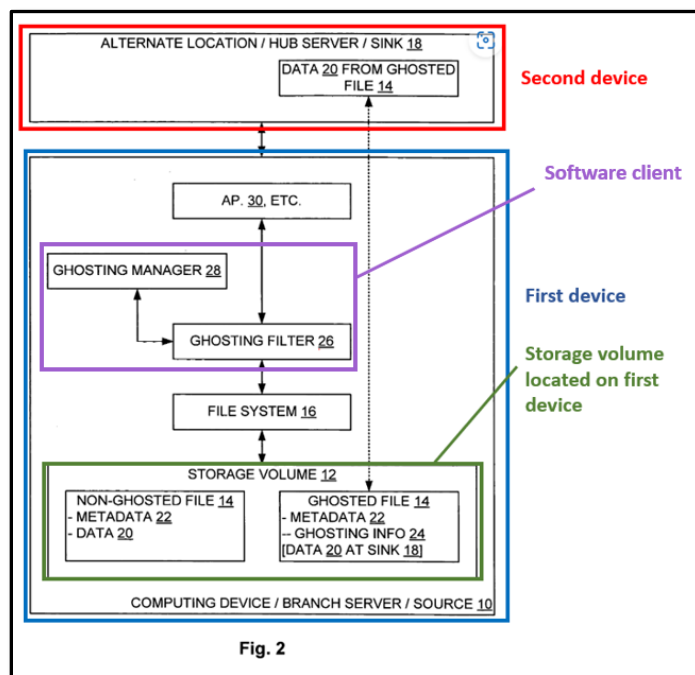
As such, Havewala discloses “wherein a visual representation of the singular file system on the first device [e.g., the file system including its files on the branch/computing device 10] is identical to a visual representation of the singular file system on the second device [e.g., the same file system including its files on the computing device 18].”

7. **1[F]: “if the file is the virtual file of the corresponding file physically located on the second device, requesting by the software client on the first device that a peer-to-peer connection be brokered by a server-based web service between the first device and the second device”**

Havewala in view of Adams discloses this limitation. Havewala's software client comprises a **ghosting filter 26** that works alongside a **ghosting manager 28** to request that the data be transferred from the **alternative sink 18 (second device)** to the **source 10 (first device)** to reconstitute the ghosted file. Ex-1005, FIG. 2; *id.*

¶73 (“[G]hosting filter 26 may interface with the ghosting manager 28 to perform higher-level ghosting functionality and **direct access to network resources such as the sink 18**, the ghosting filter 26 issues to the ghosting manager 28 a **request**

to obtain some portion of the data 20 for the ghosted file 14 [or the entire non-ghosted file] from the sink 18”).



Ex-1005, FIG.2

Havewala discloses that the ghosting manager 28 “includes all necessary functionality” to obtain the corresponding physical file stored on a second device (sink 18). Ex-1005, ¶73. For example, Havewala discloses that the data or file “can then be moved from the sink 18 to the source 10 **by way of an appropriate transport mechanism and conduit.**” *Id.*, ¶60; *see also id.*, ¶55. The network described in Havewala is a “wide area network (WAN) 152” including “networking environments [] commonplace in offices, enterprise-wide computer networks, intranets, and the Internet.” *Id.*, ¶39. A PHOSITA would have

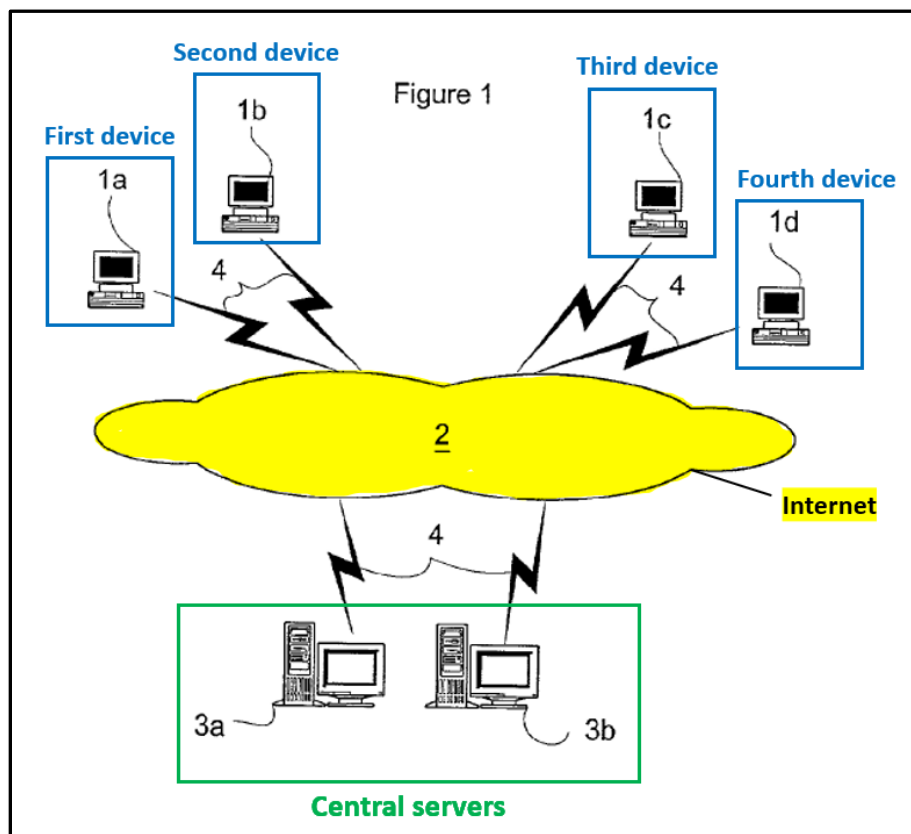
understood that common transport mechanisms used for the Internet at the time included peer-to-peer connections brokered by a server-based web service, such as that disclosed by Adams. Ex-1003, ¶¶142-148.

In particular, Adams discloses a “second client terminal” that is “remote to the first client terminal” and that “the second client terminal **mak[es] a peer-to-peer network connection** to the first client terminal to obtain [a] selected file[,] and subsequent[ly]...download[s the file] to the second client terminal.” Ex-1006, ¶25. Adams explain that its “[p]eer-to-peer connections are made either via a direct TCP/IP connection, or via a relay **server** that passes information between nodes on the network.” *Id.*, ¶51.

The term “web service” does not require construction because the ’260 patent clearly describes “web service” as “a part of the system that **operates on the Internet** and is made up of several types of web technologies including web **servers**, databases, **peer-to-peer services** (e.g., jingle, BitTorrent, Skype, JXTA), an HTTP Relay and other systems that **interact with the clients**.” Ex-1001, 4:48-52. In other words, as used in the ’260 patent, peer-to-peer services are “web services.”

As shown in Figure 1, Adams’s “**client terminals 1**” and “**central index servers 3**” are “connected to the **internet 2**.” *Id.*, ¶33. And Adams’s “**peer-to-peer network connection**” between the first and second terminals is brokered through a

“**central index server**” which connects devices through a TCP/IP connection. *Id.*, ¶¶25, 51; Ex-1003, ¶147. Adams further discloses that the “software on User B’s computer [] makes a peer-to-peer network connection to User A’s computer to obtain the file.” Ex-1006, ¶51.



Ex-1005, FIG.1.

A PHOSITA would have understood that Adams discloses a server-based web service because Adam discloses explicitly a server that addresses limitations of search engines available at the time that attempted to locate Internet websites of interest based on search terms (Ex-1006, ¶10) and that users could manually search

for or browse files via the central index server, whose results contain **links** to download files. *Id.* ¶¶15-16 (Adams discussing “search results list contain[s] a **link** to at least one file which is located on the further client terminal....”).

A PHOSITA would have understood Adam’s **links** to be web-based. Notably, “links” is shorthand for “hyperlinks” (such as a uniform resource locator or “URL”) and in the context of “searching for files” in a distributed file system, a link is a reference to an address on the web. *See* Ex-1028. In other words, Adams not only discloses a peer-to-peer connection—which, per the ’260 patent is a web service—but it also discloses that the service uses links or web addresses to remote files, further confirming a web service as defined by the ’260 patent. Ex-1003, ¶145.

Moreover, a PHOSITA would have recognized that Adams’s central index server brokers a peer-to-peer connection between the first and second client terminal because a PHOSITA would understand that brokering a connection simply means assisting in establishing a peer-to-peer session (e.g., by identifying available end point client terminals). Ex-1003, ¶147. Adams discloses such brokering because “the location of the selected file [is] transmitted **from the central index** server to the second client terminal.” Ex-1006, ¶25. As another example, Adams discloses that the “central index server 3 returns a list of files 6 and **file locations**” and, based on the server’s search results, client terminal 1 “can

retrieve a file 6 from the search results by **requesting it from one of the client terminals 1 specified in the search results (a peer-to-peer transfer).**” *Id.*, ¶37; Ex-1003, ¶146.

Therefore, Havewala in view of Adams discloses **“if the file is the virtual file of the corresponding file physically located on the second device [e.g., Havewala’s ghosted file on the sink device], requesting by the software client on the first device that a peer-to-peer connection be brokered by a server-based web service between the first device and the second device [e.g., Havewala’s source device modified by Adams to request a peer-to-peer connection be brokered by Adam’s central index server connected to the internet between the source and the sink in order to obtain the data from the non-ghosted file on the sink].”**

8. 1[G]: “if the peer-to-peer connection is brokered, transferring the corresponding physical file from the second device to the first device”

As discussed in Section IX(A)(7), which is fully incorporated herein, it would have been obvious to a PHOSITA to combine the teachings of Havewala and Adams to broker a **peer-to-peer connection** by a **server-based web service** to **transfer the physical file from the second computing device to the first computing device.**

Transfer of physical file from the second to the first device. Havewala in view of Adams discloses transferring a physical file from the second computing

device to the first computing device. *First*, while claim 1 requires transferring a physical file from the second device to the first device, it does **not** require sending a file containing both the data and the metadata. Havewala clearly discloses transferring the data from the second device to the first device, and a PHOSITA would understand that this data is transferred as a physical file. *Second*, to the extent it is argued that the claim somehow requires sending a physical file that includes both the data and the metadata (despite this not being required by the claim), Adams discloses transmitting such a file, and it would have been obvious for a PHOSITA to modify Havewala to transfer such a physical file. Addressing each of these in turn:

First, Havewala Transmits Data 20. As discussed in Section IX(A)(5), Havewala discloses that, after a non-ghosted file corresponding to a ghosted file is located (e.g., through the ghosting information metadata associated with the ghosted file), the physical file (e.g., containing the primary data 20) is “**moved from the sink 18 to the source 10,**” to “reconstitute” the ghosted file (*i.e.*, make the virtual ghosted file a physical non-ghosted file). Ex-1005, ¶¶60, 74, 75, FIG. 6; Ex-1003, ¶¶152-153. When Havewala transfers the primary data 20 from the sink to the source, Havewala transfers the data (e.g., Ex-1005, ¶52 (“primary data is the data 20 from the file 14 as may be employed by an application 30”)) but does not send the metadata and the alternate data (which Havewala states could include a

thumbnail used by the file system), and updates the metadata information to reflect the transfer (e.g., *id.*, ¶61 (“the metadata 22 for the no-longer-ghosted file 14 is amended to remove the ghosting information 24”).

This is the same as what the '260 patent states is transferred—file content, not all associated metadata. *See* Ex-1001, 6:67-7:6 (discussing removing a “file from memory, so that it is stored only in the single location/device” and that the “file’s metadata is updated...and propagated to all other devices”), 6:9-11 (distinguishing “each identified data file” from “the meta-index [that] includes metadata” about the file). A PHOSITA would understand that the data is transferred as a physical file that is then reconstituted and opened. Ex-1003, ¶153.

Second, to the extent that PO argues this limitation requires sending an entire file (including the data and the metadata) and that Havewala does not disclose or render obvious this limitation, Adams discloses it. Havewala discloses that in some embodiments “the entire file is stored at the sink,” (*i.e.*, Havewala sends the entire file from the source to the sink). Ex-1005, ¶24. As shown in Figure 3, the entire file (e.g., non-ghosted file) resides at the sink and a non-ghosted file contains metadata, primary data, and alternate data. *Id.*, FIG. 3b. Therefore, Havewala discloses transferring entire files (data and metadata) between computing devices.

Moreover, Adams discloses that a “client terminal 1a...can **retrieve a**

file...by requesting it from the client terminal 1c.” Ex-1006 ¶39; *see also id.* ¶23 (“[T]he selected **file** then being accessed by or **downloaded to the first client terminal from the further client terminal.**”). In other words, just like in Havewala, Adams discloses transferring an entire file between computing devices. In Adams, only entire files are transferred, not in some instances only the data as in Havewala.

To the extent transferring a physical file requires transferring the entirety of the file, including the metadata (and that this is not already disclosed or rendered obvious by Havewala), a PHOSITA implementing Havewala would also be motivated to incorporate Adam's disclosure of transferring entire files for at least the reasons discussed in Section XII(A)(1)(h) above. For example, a PHOSITA would understand that both: (1) there is value in transferring the metadata in the file (discussed subsequently), and (2) metadata is on the order of 1024 bytes for NTFS (the standard file system used in Windows), and thus the overhead in transferring the metadata is unarguably inconsequential (consider that a single TCP packet would likely be sufficient to send the entire metadata in that payload data can be up to roughly 1500 bytes for both ethernet and WIFI communication links).

Moreover, a PHOSITA would see the value in transferring metadata along with the file for at least the following additional reasons: file metadata contains (1) timestamp information, which can be used as a straightforward conflict-resolution

mechanism (which is disclosed by Havewala, *e.g.*, Ex-1005, ¶127) to determine which data should be considered the “latest” in scenarios where the source and the sink both modify the file, (2) size information, which can be used to inform sources if the file size on the sink exceeds the current storage capacity of the source (a problem directly emphasized by Havewala), (3) file attributes, which inform the file system if the file is actually an executable program or should be read-only, and (4) permissions, which can be used to determine if the source would even have the necessary access rights to retrieve the data from the sink.

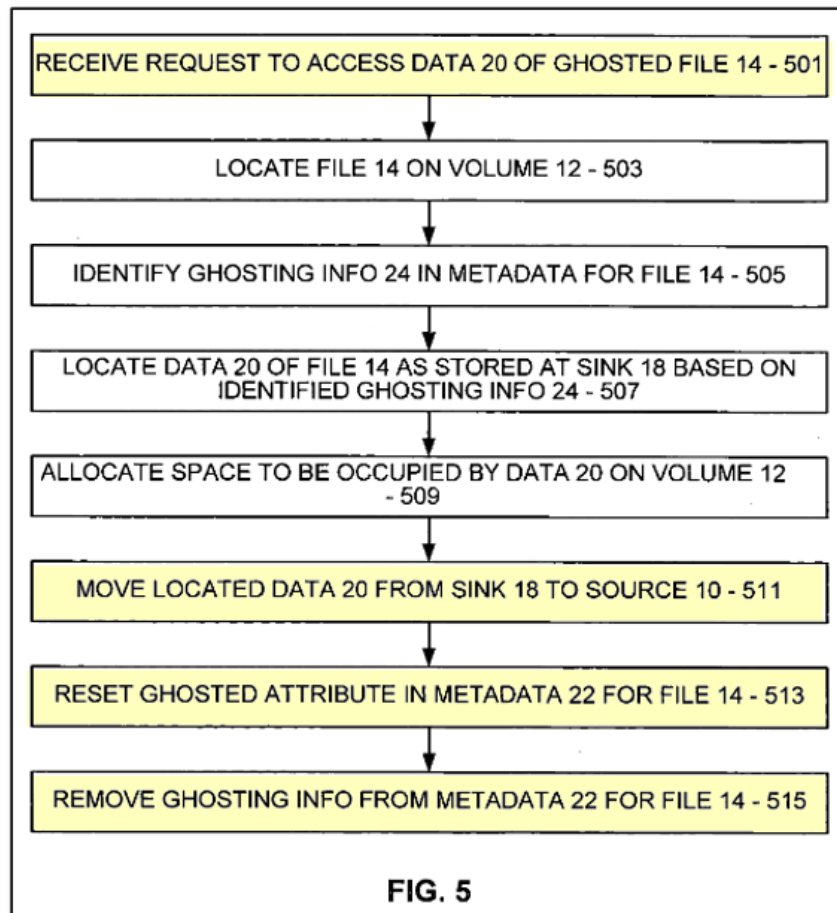
Transfer After Peer-to-peer connection brokered by server-based web service. Adams discloses that after the central index server transmits “the location of the selected file...to the second client terminal,” the “second client terminal mak[es] a peer-to-peer network connection to the first client terminal to **obtain the selected file**” and the file is “**downloaded to the second client terminal.**” Ex-1006, ¶25. It is inherent, *i.e.*, necessarily required, that the peer-to-peer connection is brokered or established before the physical file is transferred. Ex-1003, ¶158.

Therefore, Havewala in view of Adams discloses “**if the peer-to-peer connection is brokered** [e.g., once Adam’s peer-to-peer connection is brokered between Havewala’s source and sink], **transferring the corresponding physical file from the second device to the first device** [e.g., Havewala’s non-ghosted data is sent from the sink to the source to reconstitute the ghosted file, or modified by

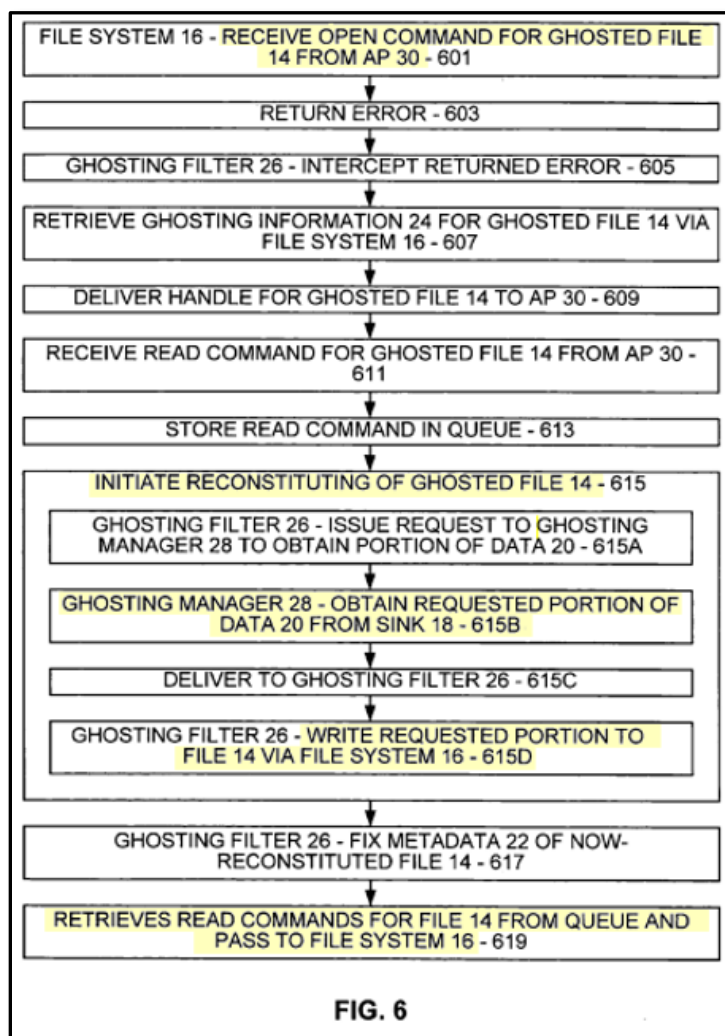
Adams to send the entire physical file from Havewala's sink to its source]."

9. 1[H]: "performing the operation on the transferred corresponding physical file at the first device."

Havewala in view of Adams discloses this limitation. Havewala's operations, for example an open or read command, can be performed on the physical file once back at the source 10 (first device). *See* Ex-1005, ¶64 ("[O]perations of FIG. 5 [] are performed at the source 10"), ¶66 ("[A]pplication 30...issu[es] an open command to the file system 16...then issu[es] a read command to the file system 16 with regard to such opened file 14), FIG. 6.



Ex-1005, FIG.5.



Ex-1005, FIG.6.

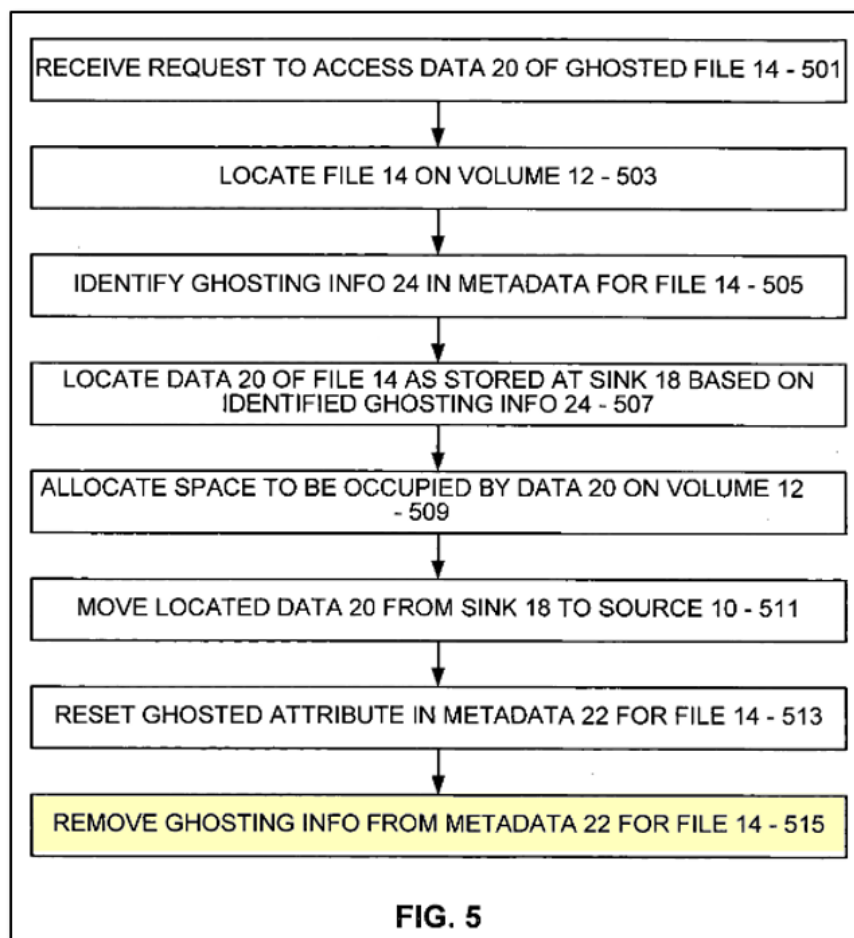
Once the physical file is transferred from the second device back to the first device, Havewala’s ghosting filter passes operation commands to the first device’s file system to perform on the physical file. *See id.* ¶¶71, 76.

Therefore, Havewala discloses “**performing the operation on the transferred corresponding physical file** [*e.g.*, accessing, opening, reading the file] **at the first device** [*e.g.*, source 10].”

B. Claim 4

Claim 4 depends from claim 1, which is rendered obvious by Havewala in view of Adams as discussed above and incorporated herein. Claim 4 recites “further comprising updating metadata of the singular file system on the first device and the second device to reflect a change in physical location of the file from the second device to the first device.”

As discussed in detail above in Section IX(A)(3), a ghosted file is reconstituted at the first device when the non-ghosted file or its data is transferred to the first device from the second device. *See id.* ¶¶51, 102. Havewala discloses updating the metadata across the first and second devices to reflect changes in physical locations of the ghosted and non-ghosted files, including that “the metadata 22 for the **no-longer-ghosted file 14** is **amended to remove the ghosting information 24** (step 515)” as shown in Figure 5. Ex-1005, ¶61, *see also id.*, Claim 17 (“[A]mending the metadata of the reconstituted file to remove the ghosting information”). Such ghosting information includes the location of a ghost file’s corresponding physical file because it “includes information that may be employed to retrieve the data 20 for the file 14 from the sink 18...[and] may include the ID stored with and **identifying the data 20 at the sink.**” Ex-1005, ¶58.



Ex-1005, FIG.5.

A PHOSITA would have understood that by removing the ghosting information from the no-longer ghosted (i.e., reconstituted file), the metadata is updated to reflect that the physical file is no longer located on the second device but is now local on the first device. *See id.* ¶¶51, 58, 61, 80; Ex-1003, ¶165. Specifically, metadata can be managed by the file system, and in this context, updating the metadata of the file system refers to updating (by the file system) the metadata associated with the (no-longer ghosted) file at the source. Ex-1003, ¶165,

Ex-1010. For example, in NTFS, the primary file system of the Windows operating system, all file metadata is stored in a Master File Table (MFT) and updated by the file system. Ex-1003, ¶165; Ex-1011. Moreover, the metadata of the file system at the sink (*i.e.*, second device) is also updated to reflect that the physical file is no longer located at the sink. *See* Ex-1005, ¶61, 82.

As such, Havewala in view of Adams renders obvious **“updating metadata of the singular file system on the first device and the second device [e.g., removing ghosting information in the metadata for the now non-ghosted file] to reflect a change in physical location of the file from the second device to the first device [e.g., after the file state changes on the source from ghosted state to non-ghosted].”**

C. Claim 5

Claim 5 depends from claim 1, which is rendered obvious by Havewala in view of Adams as discussed above and incorporated herein. Claim 5 recites “wherein an operating system of the first device is unaware of files being virtual or physical and believes that all files are physically stored on the first device within the singular file system.”

Havewala discloses applications that are unaware of whether a file is virtual or physical. Ex-1005, ¶¶64, 67. When attempting to operate on a virtual file, the process of retrieving the corresponding the physical file **“may be performed**

transparently” to the application. *Id.* ¶64; *see also id.*, ¶68 (“[T]he ghosting filter 26...**delivers the handle to the application 30 as the (normal) response to the open command therefrom** (step 609).”).

The '260 patent describes that when an application attempts to operate on a virtual file at the first device, “the **operating system, and in turn the requesting application** (Adobe Reader) has no idea that the request file is not local to Device A.” Ex-1001, 8:54-60. Similarly, Havewala discloses that the application believes that all files are physically stored on the first device within the singular file system because “application 30 in issuing such open command **is not expected to have any knowledge that the particular file 14 is in fact ghosted.**” Ex-1005, ¶67; *see also id.* ¶68. Further, as discussed above, the ghosted and non-ghosted files are indistinguishable to the file system. Section IX(A)(3); Ex-1005, ¶64. A PHOSITA would have understood that Havewala’s file system is the part of the operating system that deals with file organization and access, and the operating system is thus unaware of whether Havewala’s files are ghosted or non-ghosted. Specifically, the file system exposes both ghosted and non-ghosted files to the same operations (e.g., read, write, open, and close) (Ex-1005. Fig. 6, Fig 7, Fig. 9, ¶66), and thus, the operating system is not aware that the requested file is not local, until it receives an error, which is intercepted by the ghosting filter. Ex-1005, ¶64.

As such, Havewala in view of Adams renders obvious “**operating system of**

the first device [e.g., Havewala’s operating system including its file system] **is unaware of files being virtual or physical and believes that all files are physically stored on the first device within the singular file system** [e.g., file access is transparent to the operating system, file system, and applications and any file access errors are intercepted and dealt with].”

D. Claim 6

Claim 6 depends from claim 1, which is rendered obvious by Havewala in view of Adams as discussed above and incorporated herein. Claim 6 recites “further comprising: upon completing the operation, pushing the physical file back to the second device; and retaining the virtual file at the first device.”

Havewala discloses that “**a ghosted file 14 that has been reconstituted or partially reconstituted can at some point be re-ghosted.**” Ex-1005, ¶101. As discussed above with respect to claim 1, a ghosted file is “reconstituted” when its corresponding physical file is transferred from the second device (e.g., alternative location/sink/branch hub 18) to the first device (source/branch server 10) containing the ghosted file. By “re-ghosting” the reconstituted file the virtual file is retained at the first device, Havewala discloses pushing the entire physical file or data file back to the second device (i.e., sink 18). *Id.*, ¶¶80, 101, 102. Havewala teaches that the first device (source 10) is set to re-ghost reconstituted files and offload them from the first device in certain situations. *See id.*, ¶102 (“re-ghosting

is **triggered by some** event”). For instance, “possible selection factors for determining whether to **re-ghost** a particular file includes...whether the file 14 has been **modified**,” (i.e., once a file modification is complete) at the source. *Id.*

¶¶102-129. Havewala also discloses that a triggering event (e.g., lapse of time, file is too large for the storage space, or manual trigger by user) will re-ghost the selected modified file at the source. *Id.*

A PHOSITA would further recognized that Havewala’s re-ghosting by pushing the physical file back to the sink would not be necessary if the physical file remained at the sink (*i.e.*, the sink does not keep the physical file after transferring it to the source). Ex-1003, ¶174.

As such, Havewala in view of Adams renders obvious “**upon completing the operation** [e.g., after completing the access to the file], **pushing the physical file back to the second device** [e.g., re ghosting the file by sending it back to the sink]; **and retaining the virtual file at the first device** [e.g., and retaining the ghosted file at the source].”

E. Motivation to Combine Havewala and Adams

As a threshold matter, Havewala and Adams are **analogous art**, at least because they are in the **same field of endeavor** as the ’260 patent, e.g., file storage across multiple devices. The ’260 patent and both references are in CPC class G06F and U.S. class 707. All three references disclose distributed file systems for

storing files across multiple computing devices. *See, e.g.*, Ex-1001, Abstract; Ex-1005, ¶36, (“[F]ile 14 is maintained on the volume 12 of the branch server 10 in a reduced format without at least some data 20 thereof. Here such data is stored at an alternate location 18 such as a centralized hub server”); Ex-1006, ¶15 (“[T]he present invention provides a system for allowing a user to search for and locate a file in a distributed file sharing computer network”). Moreover, all address similar problems associated with accessing files that are stored across multiple devices in distributed file systems. Ex-1001, 2:7-12 (“[T]here is a need in the art for a system and process that facilitates access to content that physically resides on any one of multiple user devices...without the need to physically store all content on every device.”); Ex-1005, ¶12 (“[I]t would be useful to be able to ‘ghost’ the non-relevant files at the branch server of the particular branch...”); Ex-1006, ¶¶13-14 (related to “selectively facilitate the locating of any specific type of file...relevant to a user’s request” and identifying “a need for a new type of system for and method of distributed file sharing over a computer network.”).

Havewala discloses that any suitable transport mechanism known in the art may be used to transfer the physical file from the sink to the source. Ex-1005, ¶¶60. A PHOSITA looking to implement Havewala’s system would have been motivated to look to other references suitable transport mechanisms for transferring files within distributed file systems, including Adams’ peer-to-peer

connections that are also directed to a distributed file system focused on efficiently locating a particular file on a remote device and quickly and securely transferring the file from the remote device to a local device. *See* Ex-1006, ¶¶10-13, 26; Ex-1003, ¶¶176-179.

A PHOSITA would have had a **reasonable expectation of success** in incorporating Adams’s peer-to-peer connections into Havewala’s system. Prior to the Critical Date, a PHOSITA would have understood that there was a finite number of transport mechanisms for transferring files, and that a peer-to-peer connection was one of the most widely used methods for connecting computing devices. It further would have been simple for the PHOSITA to implement Adam’s central index server to broker connections between devices and establish “[p]eer-to-peer connections [that are] made...via a relay server that passes information between nodes on the network.” Ex-1005, ¶51. It would have also been readily apparent to a PHOSITA that Adam’s “central index servers 3 are also connected to the Internet 2” (Ex-1006, ¶33), confirming and leading a PHOSITA to the simple implementation of brokering peer-to-peer connections through a server-based web service. Ex-1003, ¶178. Peer-to-peer networks were increasingly popular with the advent of broadband in the early 2000s in that they substantially reduced the high load placed on a central server by offloading work directly between peers. Ex-1003, ¶178, Ex-1012. Moreover, it would have been

obvious to a PHOSITA to try to implement Adams system into Havewala, including because there were a limited number of ways to transport files between devices in a distributed file system.

To the extent transferring a physical file requires transferring the entirety of the file, including the secondary data (and that this is not already disclosed or rendered obvious by Havewala), a PHOSITA implementing Havewala would also have been motivated to incorporate Adams's disclosure of transferring entire files for at least the reasons discussed in Section XII(A)(1)(h) above. Ex-1003, ¶179. For example, a PHOSITA would Understand that receiving the entire file including the metadata would be advantageous for detecting changes in the file or its access permissions. A PHOSITA would have had a reasonable expectation of success because this would have been a modification. *Id.*

F. Claim 7

1. **7[P]: “A non-transitory computer-readable storage medium storing a set of instructions that, when executed by a processor, cause the processor to perform operations, comprising:”**

To the extent the preamble is limiting, Havewala discloses it.

Havewala discloses that the invention is practiced on a “general purpose computing system include[ing] a conventional personal computer 120” (Ex-1005, ¶35) which “may further include...**drives** and their associated **computer-readable**

media provid[ing] non-volatile storage of **computer readable instructions**, data structures, program modules and other data for the personal computer 120” (*id.*, ¶36); *see also id.*, ¶34 (“[T]he invention described in the general context of **computer-executable instructions**, such as **program modules**, being executed by a computer.”).

Therefore, Havewala discloses “a non-transitory computer-readable storage medium storing a set of instructions that, when executed by a processor, cause the processor to perform operations.”

2. 7[A]-[H]

As shown in the table below, claim limitations 7[A]-[H] are identical to the limitations of 1[A]-1[H], respectively, and disclosed for the same reasons discussed above and incorporated herein by reference. *See* Sections IX(A)(2)-(9); Ex-1003, ¶185.

Claim 7	Claim 1	Sections
7[A]	1[A]	IX.A.2
7[B]	1[B]	IX.A.3
7[C]	1[C]	IX.A.4
7[D]	1[D]	IX.A.5
7[E]	1[E]	IX.A.6
7[F]	1[F]	IX.A.7

7[G]	1[G]	IX.A.8
7[H]	1[H]	IX.A.9

G. Claim 8

1. 8[P]: “A client comprising:”

To the extent the preamble is limiting, Havewala discloses it.

Havewala discloses that its invention “is described in the general context of computer-executable instructions, such as program modules, being executed by a computer, such as a **client workstation** or a server.” Ex-1005, ¶34. As discussed above, Havewala’s system is installed on a first and second computing (or client) device.

Therefore, Havewala discloses a client for practicing the invention.

2. 8[A]: “a memory;”

Havewala discloses that its invention is practiced on a “general computing system include[ing]...a **system memory** 122” with the system memory “include[ing] **read-only memory** (ROM) 124 and **random access memory** (RAM) 125.” Ex-1005, ¶35. This memory is further illustrated in FIG. 1 below.

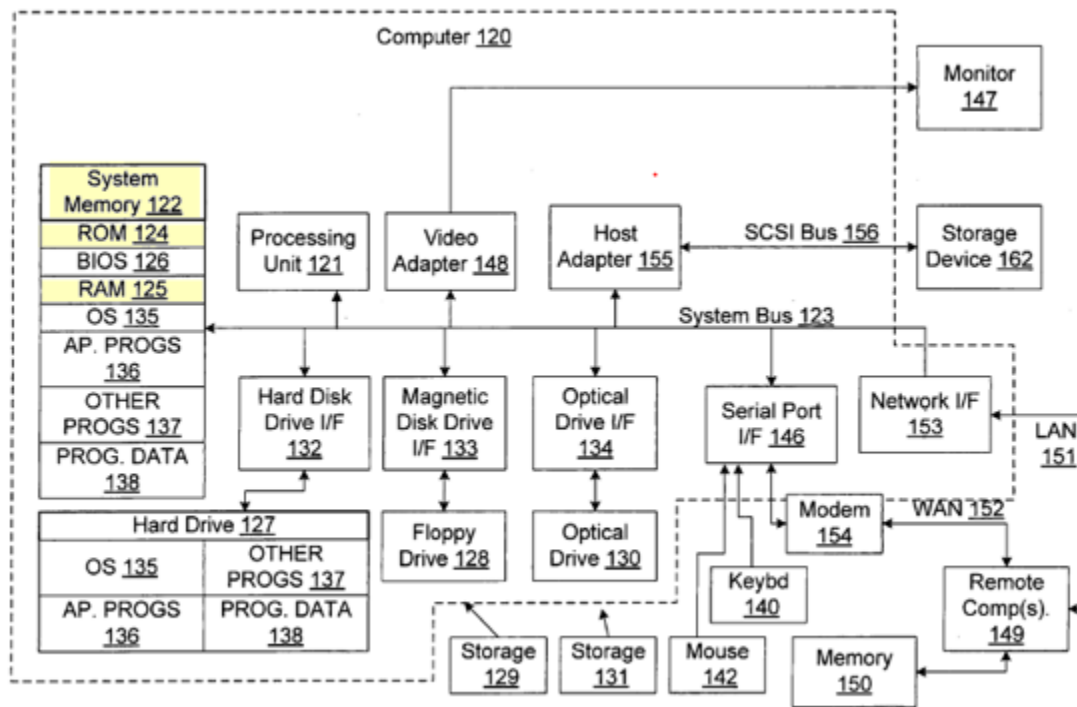


Fig. 1

Ex-1005, FIG.1.

Therefore, Havewala discloses “a memory.”

3. 8[B]: “at least one processor configured to:”

Havewala’s general purpose computing system “includ[es] a **processing** unit 121.” Ex-1005, ¶35. The general computing system provides the mechanism for implementing the Havewala system. *Id.*

Therefore, Havewala discloses “at least one processor (e.g., processing unit) configured to” practice the ’260 claims.

4. 8[C]-[J]

As shown in the table below, claim limitations 8[C]-[J] are identical to 1[A]-

1[H], respectively, and disclosed for the same reasons discussed above and incorporated herein by reference. *See* Sections IX(A)(2)-(9); Ex-1003, ¶197.

Claim 7	Claim 1	Sections
8[C]	1[A]	IX.A.2
8[D]	1[B]	IX.A.3
8[E]	1[C]	IX.A.4
8[F]	1[D]	IX.A.5
8[G]	1[E]	IX.A.6
8[H]	1[F]	IX.A.7
8[I]	1[G]	IX.A.8
8[J]	1[H]	IX.A.9

**X. GROUND 2: HAVEWALA IN VIEW OF ADAMS AND SARIDAKIS
RENDERS CLAIMS 2 AND 3 OBVIOUS**

A. Claim 2

Claim 2 depends from claim 1, which is rendered obvious by Havewala in view of Adams and Saridakis as discussed above and incorporated herein. Claim 2 further recites “**wherein if the peer-to-peer connection cannot be brokered between the first device and the second device, an alternative connection is established by the server-based web service.**” As discussed in Sections IX(A)(5) and IX(A)(3), which are fully incorporated herein, Havewala discloses maintaining

a physical file on a second device (e.g., alternate location/sink/branch hub 18) that corresponding to a ghosted file on a first device (e.g., source/branch server 18) that can be transferred to the first device “by way of an appropriate transport mechanism and conduit.” Ex-1005 ¶¶60; *see also id.* 40, 73. A PHOSITA would have understood appropriate transportation mechanisms at the time included alternative connections where a peer-to-peer connection could not be brokered. Such alternative connections are disclosed in Saridakis. Ex-1003, ¶¶198-201.

Saridakis discloses a “**peer-to-peer communication channel** between peers” that “permits the **peer device and the another peer device** to send and receive data.” Ex-1007, Abstract. Saridakis recognizes that in some instances, a peer-to-peer connection cannot be successfully established. For instance, with “strict firewall policies, there is **no straightforward way to establish a P2P connection** between two peers that lie on different side of such a firewall.” *Id.*, 1:64-67. To combat this issue, Saridakis discloses a known “method of **circumventing network obstacles** to provide a peer-to-peer communication channel between peers.” *Id.*, 3:7-9. Saridakis achieves this circumvention (i.e., alternative connection) by “a first peer device communicating with a relay via a network **including an obstacle**, a second peer device communicating with the same relay via a network including another device, and a **server** coupled to the first and second peer devices and **including programmed instructions to**

carrying out functions of relaying the communication from the first peer device to the second and vice versa.” *Id.*, 7-17. Moreover, Saridakis’s server is the claimed server-based web service because it is “coupled to the network and communicates using HTTP (hypertext transfer protocol) messages” where the “network 18 can be the Internet.” Ex-1007, 3:62-65. HTTP is a common protocol used to communicate with the World-Wide-Web (often abbreviated as, “the Web”) to retrieve documents, and thus, by definition, the HTTP Server used in Saridakis is “web-based.” Ex-1003, ¶200, Ex-1013.

Therefore, Havewala in view of Adams and Saridakis renders obvious “**if the peer-to-peer connection** [e.g., Adams’s peer-to-connection between clients via central index server] **cannot be brokered between the first device and the second device, an alternative connection is established by the server-based web service** [e.g., Saridakis’s HTTP communication].”

B. Claim 3

Claim 3 depends from claim 2, which is rendered obvious by Havewala in view of Adams and Saridakis as discussed above and incorporated herein. Claim 3 further recites “wherein the alternative connection is an HTTP relay.” As discussed in Section X(A), incorporated herein, Saridakis discloses an alternative connection to broker peer-to-peer connections as an appropriate transportation mechanism for the Havewala system.

For example, Saridakis discloses a system for establishing “**peer to peer connections** between personal computers (PCs) and smart phone **despite the obstacles imposed by firewalls**” by way of “utiliz[ing] an **HTTP-based protocol that does message relaying.**” Ex-1007, 2:48-55. The ’260 patent identifies this same reason as a reason to establish an alternative HTTP connection. *See* Ex-1001, 8:52-54 (“In instances where p-to-p connection is not possible, e.g., due to firewall restriction, the web service acts as an HTTP relay server.”). As discussed in Section X(A), Saridakis further discloses a server that corresponds to the ’260 patent’s server-based web service. Moreover, the “**server receives a HTTP request** from the first peer devices. This HTTP request includes data intended for another peer device. The server further **relays** the aforementioned data to the intended peer device **establishing thus a virtual communication channel between the first peer device and the second peer device** to enable sending and receiving of data.” Ex-1007, 3:17-23.

Therefore, Havewala in view of Adams and Saridakis renders obvious an “alternative connection [of] an HTTP relay.”

C. Motivation to Combine Havewala, Adams, and Saridakis

Havewala, Adams and Saridakis are **analogous art** at least because they are in the **same field of endeavor** as the ’260 patent, *i.e.*, sharing data across multiple devices. Like Havewala, Adams, and the ’260 patent, Saridakis is also in CPC

class G06F and discloses “send[ing] and receiv[ing] data” between devices through some form of “communication channel.” Ex-1007, Abstract.

A PHOSITA implementing the Havewala-Adams system would have noted that Havewala recognizes that the reconstitution or buffering of “a ghosted file 14 may be interrupted, for example by a loss of power or network connectivity at the source 12.” Ex-1005, ¶83. A PHOSITA would have recognized that Adams discloses that, “[p]eer-to-peer connections are made via a direct TCP/IP connection, or via a relay server that passes information between nodes on a network.” However, Adams does not provide details on how this relay server would work to support “passing information between nodes.” Notably, P2P networks became common alongside the popularity of the World-Wide-Web in the early 2000s, and thus, a PHOSITA would look to Saridakis in leveraging the Web-based services (which all clients with Internet service can connect to) to serve as the relay server in the event a direct TCP/IP connection would not be possible. Ex-1003, ¶205, Ex-1012.

A PHOSITA would have had a **reasonable expectation of success** in implementing Saridakis’ system to the Havewala-Adams combination. Each reference discloses well known components for web-traffic and modifying the Havewala-Adams combination to include Saridakis is routine and easy for a PHOSITA (or any network engineer).

XI. GROUND 3: HAVEWALA IN VIEW OF ADAMS, SARIDAKIS AND ROTHMAN RENDERS CLAIMS 1-8 OBVIOUS

A. Claims 1-8

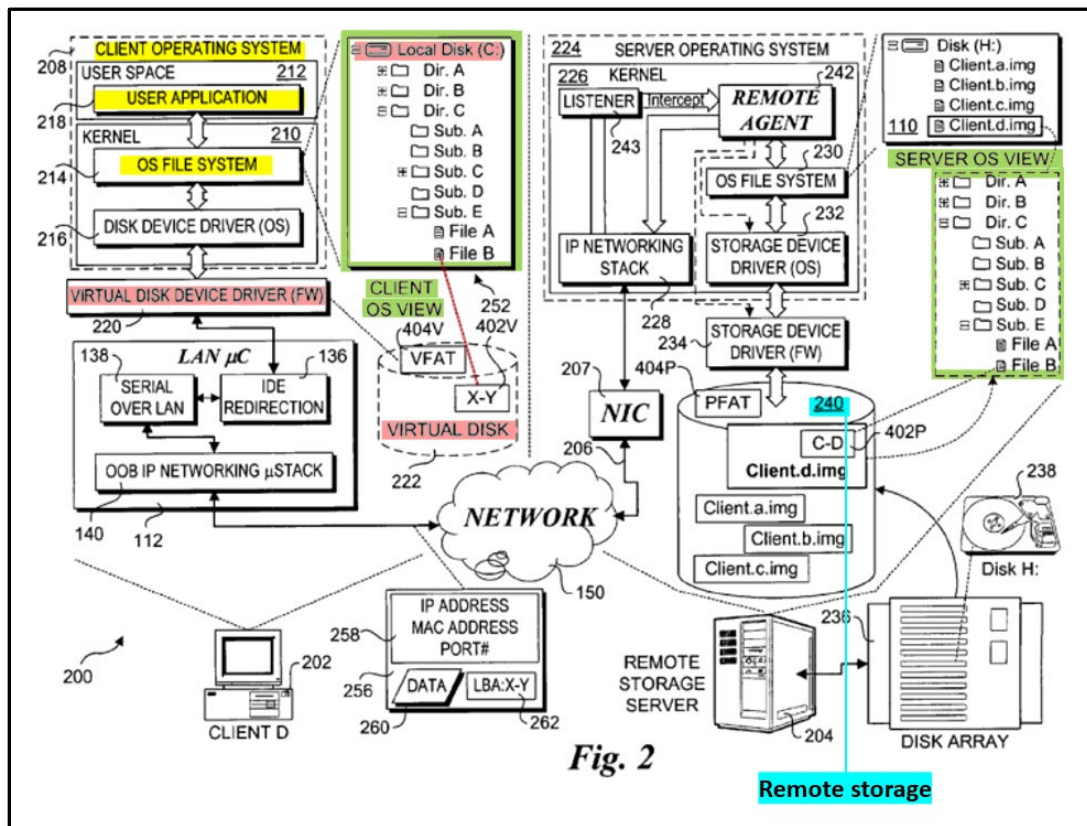
To the extent it is argued by PO that Havewala in view of Adams and Saridakis does not explicitly disclose (i) the “singular file system” recited in 1[pre], (ii) “local files and virtual files appear[ing] indistinguishable to the singular file system” recited in 1[B], and/or (iii) claim 5, Rothman discloses all. Ex-1003, ¶206-212.

Claim 1[pre] – Singular file system.

As discussed in Section IX(A)(1) (discussing limitation 1[pre]), Havewala discloses (or at least renders obvious) that its system would have been installed on multiple devices—e.g., branch servers 10 and branch hubs 18—that include the same operating systems and file systems (e.g., Windows file system as modified by Havewala). Once Havewala’s ghosting software is installed, the representations of the file system and files stored therein on both the first and second devices would appear the same. Ex-1005, ¶52; Ex-1003, ¶209.

Rothman similarly discloses a system that includes virtual and local files, “wherein the client runs an operating system that is a member of the Microsoft Windows family of operating systems, and the remote storage server runs an operating system that is also a member of the Microsoft Windows family of

operating systems”). Ex-1008, claim 12. As shown in Figure 2, with the same family of operating systema on a client (first device) and remote storage server (second device), the **file system** across both devices is the same (*i.e.*, singular file system).



Ex-1008, FIG.2.

To the extent that Patent Owner argues that Havewala does not disclose or render obvious a singular file system or how to implement or structure the singular file system across multiple computers (it does), a PHOSITA would have been motivated to look to other references concerning how to implement Havewala’s

ghosting system across a family of computers in, for example, the branch server/branch hub environment as discussed in Havewala. Ex-1005, ¶¶13-17.

A PHOSITA would have been led and motivated to apply the teachings of Rothman because it confirms that Havewala's system would be installed across the related computers utilizing the same family of standard and widely used operating and file systems (e.g., Windows and Windows file system as modified by Havewala). A PHOSITA would have also been led to Rothman because Rothman discloses an exemplary display of a file system hierarchy—e.g., Client OS View and Server OS View as shown in Figure 2—demonstrating that the file hierarchies on both devices would be the same. Rothman would further confirm to a PHOSITA implementing Havewala's system that the hierarchal structure of the file system across a family of computers should appear the same on those devices.

Claim 5 - virtual files that appear as local files to an operating system.

Rothman discloses that a “**virtual disk drive** comprising an emulation of a non-existent local (to a client) disk drive is facilitated via out-of-band (OOB) communications with a remote storage server in a manner that **is transparent to an operating system (OS) running on the client.**” Ex-1008, Abstract. Moreover, the “**virtual disk appears to client operating system 208** (and thus to all of user applications 212) **as a local disk drive.**” *Id.*, ¶39; *see also, id.*, ¶76 (“In fact, from the operating system's standpoint, the virtual disk actually exists as a local disk.”).

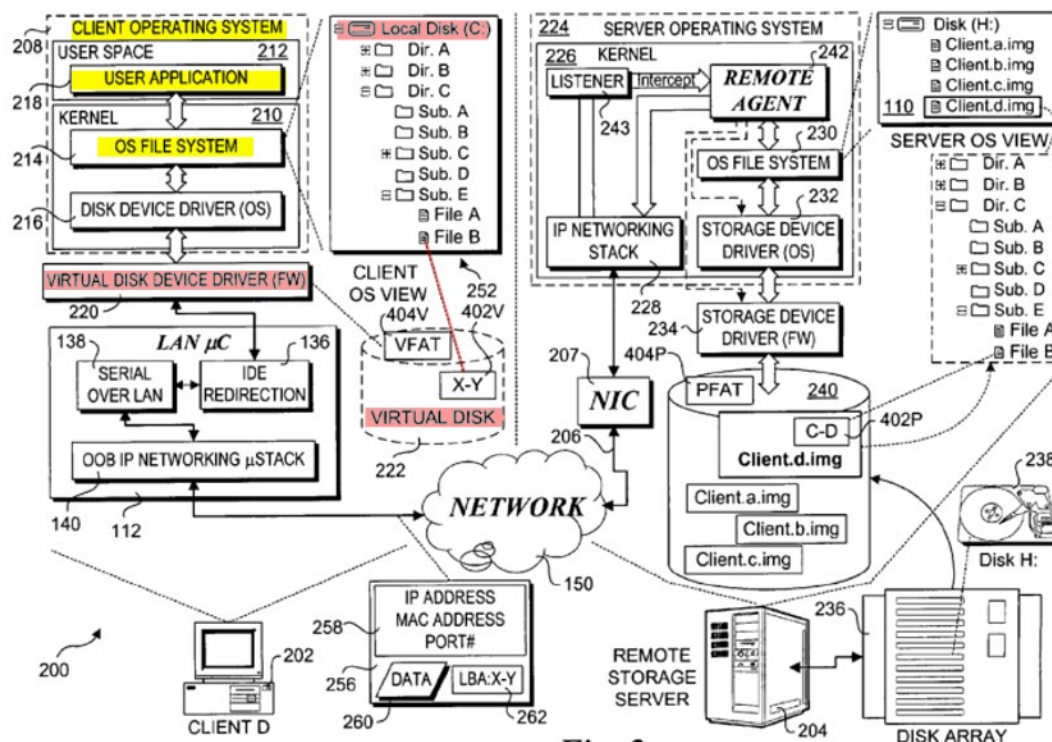


Fig. 2

Claim 1[B] - Virtual and physical files indistinguishable to the file system.

As discussed above, Rothman discloses virtual files that appear as local files to an operating system. Rothman further discloses that the file system similarly does not distinguish between virtual and local files by explaining that “among the[] core OS components is an OS file system 214.” Rothman’s virtual disk drives have “virtual file[s] ‘stored’” on the disk. Ex-1008, ¶30. Therefore, a PHOSITA would have readily understood that these **virtual files** necessarily **appear local** to the file system and operating system because they **appear as if they reside on a local disk drive** [e.g., Files A and B appear on the client, but are actually located on the server]. Ex-1003, ¶214. Specifically, the operations made by the

operating system that are applied to the virtual files that reside on the virtual disk are identical to conventional local disks. Ex-1008, ¶40. That is, traditional file system operations, such as reads and writes, can be applied to both local files and virtual files, as shown in Fig. 3a. and Fig 3b. Ex-1003, ¶213.

B. Motivation to Combine Havewala, Adams, Saridakis, and Rothman

Havewala, Adams and Rothman qualify as **analogous art** at least because they are in the **same field of endeavor** as the '260 patent, i.e., accessing remotely stored files. Like Havewala, Adams, and the '260 patent, Rothman is in CPC class G06F. Also, like Havewala, Adams, and the '260 patent, Rothman discloses “send[ing] and receiv[ing] data” between devices through some form of “communication channel.” Ex-1007, Abstract.

Havewala discloses that its “application 30 in issuing such open command **is not expected to have any knowledge that the particular file 14 is in fact ghosted.**” Ex-1005, ¶67. Havewala further discloses that this “**transparency** is likely preferable to such user or application.” *Id.*, ¶64. A PHOSITA implementing Havewala would look to other references utilizing virtual files for improvements to this preferred transparency, including Rothman. Specifically, a PHOSITA would recognize that providing transparency is preferred because applications and the operating system do not require additional functionality to, for example, open files.

Thus, this transparency increases the compatibility of the Havewala file system across a broad range of applications and is more reliable in that a software developer implementing Havewala would only have to focus on supporting a single file system, thereby reducing the complexity of the software. Ex-1003, ¶215.

Rothman discloses that its “virtual disk appears to client operating system 208 (and **thus to all of user applications 212**) as a **local disk drive**.” Ex-1008, ¶39. Specifically, virtual disks appear to applications as a local disk drive because they are often implemented as “lower-level firmware device drivers which provide a layer of abstraction” (*id.*, ¶32), wherein a consistent interface is provided across varying types of disks (*i.e.*, including local and virtual disks). Thus, this layer of abstraction means that applications (and users) can use the same familiar interface to interact with, in this context, both local and virtual disks. Thus, a PHOSITA would look to Rothman to provide a uniform interface such that whether a file is local or remote is transparent to the user, application, and operating system. Ex-1003, ¶216.

A PHOSITA would have a **reasonable expectation of success** as modifying Havewala-Adams-Saridakis to include Rothman’s added transparency is fully compatible with the Havewala-Adams-Saridakis system and easily implemented by a competent OS software engineer familiar with Microsoft Windows. Ex-1003,

¶213-216.

XII. CONCLUSION

Trial should be instituted and the Challenged Claims cancelled.

Dated: December 14, 2023

Respectfully Submitted,

/ Dion M. Bregman/

Dion M. Bregman

Reg. No. 45,645

CERTIFICATION OF COMPLIANCE WITH TYPE-VOLUME LIMITS

This Petition includes 13,989 words as counted by Microsoft Word and is therefore in compliance with the 14,000-word limit established by 37 C.F.R.

42.24(a)(1)(i). Accordingly, pursuant to 37 C.F.R. 42.24(d), lead counsel for the Petitioner hereby certifies that this Petition complies with the type-volume limits established for a petition requesting IPR.

Dated: December 14, 2023

Respectfully Submitted,

/ Dion M. Bregman/

Dion M. Bregman

Reg. No. 45,645

CERTIFICATE OF SERVICE

Pursuant to 37 C.F.R. 42.6(e)(4) and 42.105, lead counsel for Petitioner hereby certifies that on December 14, 2023, copies of this Petition and all supporting exhibits were sent via Priority Mail Express to the correspondence address of record for the '260 patent:

Schwegman Lundberg & Woessner, P.A.
P.O. Box 2938
Minneapolis, MN 55402

Courtesy copies have also been delivered electronically to counsel of record in the parallel District Court proceeding:

Skiermont Derby LLP
Alexander Gasser
agasser@skiermontderby.com
Jaime Olin
jolin@skiermontderby.com
John Lord
jlord@skiermontderby.com
Ryan Hargrave
rhargrave@skiermontderby.com
Paul Skiermont
pskiermont@skiermontderby.com

Dated: December 14, 2023

Respectfully Submitted,

/ Dion M. Bregman/

Dion M. Bregman

Reg. No. 45,645

Appendix A

U.S. PATENT NO. 8,484,260 – Claim Listing

Claims 1	Claim 7	Claim 8
1[P]: A process for operating on files located on multiple devices using a singular file system comprising:	—	—
—	7[P]: A non-transitory computer-readable storage medium storing a set of instructions that, when executed by a processor, cause the processor to perform operations, comprising:	—
—	—	8[P]: A client comprising:
—	—	8[B]: a memory;
—	—	8[C]: at least one processor configured to:
1[A]: accepting a request to operate on a file at a first device, wherein the file is selected from the singular file system on the first device;	7[A]: accepting a request to operate on a file at a first device, wherein the file is selected from the singular file system on the first device;	8[D]: accept a request to operate on a file at a first device, wherein the file is selected from the singular file system on the first device;
1[B]: modifying the singular file system on the first device to make local files and virtual files appear indistinguishable to the singular file system, the local files and virtual files sharing a same location on the first device;	7[B]: modifying the singular file system on the first device to make local files and virtual files appear indistinguishable to the singular file system, the local files and virtual files sharing a same location on the first device;	8[E]: modify the singular file system on the first device to make local files and virtual files appear indistinguishable to the singular file system, the local files and virtual files sharing a same location on the first device;

1[C]: intercepting the request by a software client on the first device;	7[C]: intercepting the request by a software client on the first device;	8[F]: intercept the request by a software client on the first device;
1[D]: determining by the software client if the file is physically located on the first device or if the file is a virtual file of a corresponding file physically stored on a second device by reviewing file metadata,	7[D]: determining by the software client if the file is physically located on the first device or if the file is a virtual file of a corresponding file physically stored on a second device by reviewing file metadata,	8[G]: determine by the software client if the file is physically located on the first device or if the file is a virtual file of a corresponding file physically stored on a second device by reviewing file metadata,
1[E]: wherein a visual representation of the singular file system on the first device is identical to a visual representation of the singular file system on the second device; and	7[E]: wherein a visual representation of the singular file system on the first device is identical to a visual representation of the singular file system on the second device; and	8[H]: wherein a visual representation of the singular file system on the first device is identical to a visual representation of the singular file system on the second device; and
1[F]: if the file is the virtual file of the corresponding file physically located on the second device, requesting by the software client on the first device that a peer-to-peer connection be brokered by a server-based web service between the first device and the second device;	7[F]: if the file is the virtual file of the corresponding file physically located on the second device, requesting by the software client on the first device that a peer-to-peer connection be brokered by a server-based web service between the first device and the second device;	8[I]: if the file is the virtual file of the corresponding file physically located on the second device, request by the software client on the first device that a peer-to-peer connection be brokered by a server-based web service between the first device and the second device;
1[G]: if the peer-to-peer connection is brokered, transferring the corresponding physical file from the second device to the first device; and	7[G]: if the peer-to-peer connection is brokered, transferring the corresponding physical file from the second device to the first device; and	8[J]: if the peer-to-peer connection is brokered, transfer the corresponding physical file from the second

		device to the first device; and
1[H]: performing the operation on the transferred corresponding physical file at the first device.	7[H]: performing the operation on the transferred corresponding physical file at the first device.	8[J]: perform the operation on the transferred corresponding physical file at the first device.

No.	Claim Elements
Claim 2	2. The process according to claim 1, wherein if the peer-to-peer connection cannot be brokered between the first device and the second device, an alternative connection is established by the server-based web service.
Claim 3	3. The process according to claim 2, wherein the alternative connection is an HTTP relay.
Claim 4	4. The process according to claim 1, further comprising updating metadata of the singular file system on the first device and the second device to reflect a change in physical location of the file from the second device to the first device.
Claim 5	5. The process according to claim 1, wherein an operating system of the first device is unaware of files being virtual or physical and believes that all files are physically stored on the first device within the singular file system.
Claim 6	6. The process according to claim 1, further comprising: upon completing the operation, pushing the physical file back to the second device; and retaining the virtual file at the first device.