

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re <i>Ex Parte</i> Reexamination of:	)	
	)	
U.S. Patent No. 7,120,832	)	
	)	
Issued: October 10, 2006	)	Group Art Unit: To Be Assigned
	)	
Named Inventor: Kevin Collins	)	Examiner: To Be Assigned
	)	
Control Number: To Be Assigned	)	
	)	
Filed: September 27, 2001	)	
	)	
Title: STORAGE DEVICE PERFORMANCE	)	<u>VIA EFS-Web</u>
MONITOR	)	
	)	
	)	
	)	

Mail Stop *Ex Parte* Reexam
Attn: Central Reexamination Unit
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Commissioner:

REQUEST FOR *EX PARTE* REEXAMINATION

Requester NetApp, Inc. respectfully requests *ex parte* reexamination of U.S. Patent No. 7,120,832, assigned to Patent Owner's Valtrus Innovations Ltd., per the assignments at reel/frame: 056157/0492. Prior art not cited or applied during prosecution renders claims 1-26 unpatentable.

As this Request demonstrates, prior art references, including at least *Schibilla, Nagar, Hospodor, Hetzler, Jensen, Voigt, Wolf, Christenson*, and *Georgiou* raise substantial new questions of patentability ("SNQs") with respect to claims 1-26. Indeed, this prior art renders

these claims unpatentable under 35 U.S.C. § 103. NetApp thus respectfully requests that the Office grant this Request for *ex parte* reexamination of the '832 patent and cancel claims 1-26.

In accordance with 37 C.F.R. § 1.510(b), this Request includes the following:

- (1) A statement pointing out each substantial new question of patentability based on prior patents and printed publications.
- (2) An identification of every claim for which reexamination is requested, and a detailed explanation of the pertinency and manner of applying the cited prior art to every claim for which reexamination is requested. For each statement of the patent owner and accompanying information submitted pursuant to [37 C.F.R.] § 1.501(a)(2) which is relied upon in the detailed explanation, the request must explain how that statement is being used to determine the proper meaning of a patent claim in connection with the prior art applied to that claim and how each relevant claim is being interpreted. If appropriate, the party requesting reexamination may also point out how claims distinguish over cited prior art.
- (3) A copy of every patent or printed publication relied upon or referred to in paragraph (b) (1) and (2) of this section accompanied by an English language translation of all the necessary and pertinent parts of any non-English language patent or printed publication.
- (4) A copy of the entire patent including the front face, drawings, and specification/claims (in double column format) for which reexamination is requested, and a copy of any disclaimer, certificate of correction, or reexamination certificate issued in the patent. All copies must have each page plainly written on only one side of a sheet of paper.
- (5) A certification that a copy of the request filed by a person other than the patent owner has been served in its entirety on the patent owner at the address as provided for in [37 C.F.R.] § 1.33(c). The name and address of the party served must be indicated. If service was not possible, a duplicate copy must be supplied to the Office.
- (6) A certification by the third party requester that the statutory estoppel provisions of 35 U.S.C. [§] 315(e)(1) or 35 U.S.C. [§] 325(e)(1) do not prohibit the requester from filing the *ex parte* reexamination request.

Attachments

- (1) Certificate of Service on Patent Owner.
- (2) Information Disclosure Statement Form PTO/SB/08.

Exhibits

- Ex. A: U.S. Patent No. 7,120,832 to Kevin Collins (“the ’832 patent”)
- Ex. B: Patent Prosecution History of U.S. Patent No. 7,120,832
- Ex. C: Declaration of P. Clark, D.Sc.
- Ex. D: MARC Entry for *Nagar*
- Ex. E: Curriculum Vitae of P. Clark
- Ex. F: Declaration of Ms. Shauna L. Wiest

Prior Art Exhibits

- Ex. PA-1: U.S. Patent No. 6,384,999 to Mark Schibilla (“*Schibilla*”)
- Ex. PA-2: Rajeev Nagar, *Windows NT File System Internals* (O’Reilly & Associates, Inc., 1st ed. September 1997) (“*Nagar*”)
- Ex. PA-3: U.S. Patent No. 4,851,998 to Andrew Hospodor (“*Hospodor*”)
- Ex. PA-4: U.S. Patent No. 5,682,273 to Stephen Hetzler (“*Hetzler*”)
- Ex. PA-5: U.S. Patent No. 5,930,828 to Craig Jensen et al (“*Jensen*”)
- Ex. PA-6: U.S. Patent No. 5,623,598 to Douglas Voigt et al (“*Voigt*”)
- Ex. PA-7: U.S. Patent No. 6,370,614 to Deniz Teoman et al. (“*Teoman*”)
- Ex. PA-8: U.S. Patent No. 6,463,509 to Deniz Teoman et al. (“*Teoman509*”)
- Ex. PA-9: U.S. Patent No. 7,150,018 to David Golds et al. (“*Golds*”)
- Ex. PA-10: U.S. Patent No. 6,460,151 to Alan Warwick et al. (“*Warwick*”)
- Ex. PA-11: U.S. Patent No. 5,893,139 to Tadanobu Kamiyama (“*Kamiyama*”)
- Ex. PA-12: U.S. Patent Publication No. 2002/0162057 to Nisha Talagala (“*Talagala*”)
- Ex. PA-13: U.S. Patent No. 5,727,144 to James Thomas Brady et al. (“*Brady*”)
- Ex. PA-14: U.S. Patent No. 5,287,363 to Paul I. Wolf et al. (“*Wolf*”)
- Ex. PA-15: U.S. Patent No. 6,324,620 to Patrick J. Christenson et al. (“*Christenson*”)
- Ex. PA-16: U.S. Patent No. 5,774,292 to Christos J. Georgiou et al. (“*Georgiou*”)

Litigation Document Exhibits

- Ex. L-1: Plaintiffs’ Complaint for Patent Infringement, *Valtrus Innovations Ltd. v. NetApp, Inc.*, No. 2:25-cv-00517-JRG, Dkt. 1 (E.D. Tex. May 9, 2025)
- Ex. L-2: Consolidation Order, *Valtrus Innovations Ltd. v. FedEx Corporation*, No. 2:25-cv-00516-JRG, Dkt. 13 (E.D. Tex. Jun. 26, 2025)
- Ex. L-3: Plaintiff’s Infringement Contentions, Appendix E4: Claim Chart for U.S. Patent 7,120,832 (E.D. Tex. Aug. 13, 2025)

TABLE OF CONTENTS

I.	RELATED PROCEEDINGS	1
A.	Certification Regarding Statutory Estoppel	1
B.	Other Proceedings	1
C.	Request for Expedited Reexamination	1
II.	IDENTIFICATION OF CLAIMS THAT SHOULD BE REEXAMINED.....	2
III.	BACKGROUND OF THE TECHNOLOGY.....	2
IV.	OVERVIEW OF THE '832 PATENT AND ITS PROSECUTION HISTORY	3
A.	Specification.....	3
B.	Claims 1-26 of the '832 Patent	5
C.	Prosecution History.....	8
D.	Level of Ordinary Skill in the Art.....	9
E.	Claim Construction for the '832 Patent.....	9
1.	“means for evaluating” (claim 22).....	10
2.	“means for enhancing” (claim 22).....	10
3.	“means for intercepting” (claim 23).....	11
4.	“means for measuring” (claim 24).....	11
V.	IDENTIFICATION OF PRIOR ART AND SUBSTANTIAL NEW QUESTIONS OF PATENTABILITY	11
A.	Identification of Prior Art.....	11
B.	Substantial New Questions of Patentability	12
1.	The Combined Teachings of <i>Schibilla</i> and <i>Nagar</i> Raise an SNQ with Respect to Claims 1, 4, 5, 7, 9, 10, 12, 13, 17-20, 22, 23, and 25.....	13
2.	The Combined Teachings of <i>Schibilla</i> , <i>Nagar</i> , and <i>Hospodor</i> Raise an SNQ with Respect to Claims 2, 3, 6, 14, 15, 24, and 26	15

3.	The Combined Teachings of <i>Schibilla</i> , <i>Nagar</i> , and <i>Hetzler</i> Raise an SNQ with Respect to Claims 8 and 18.....	16
4.	The Combined Teachings of <i>Schibilla</i> , <i>Nagar</i> , and <i>Jensen</i> Raise an SNQ with Respect to Claims 11 and 16	17
5.	The Combined Teachings of <i>Schibilla</i> , <i>Nagar</i> , and <i>Voigt</i> Raise an SNQ with Respect to Claim 21.....	18
6.	The Teachings of <i>Wolf</i> Raise an SNQ with Respect to Claims 1-4, 6, 7, and 25.....	19
7.	The Combined Teachings of <i>Wolf</i> and <i>Christenson</i> Raise an SNQ with Respect to Claims 5, 8, 10, and 26	20
8.	The Combined Teachings of <i>Wolf</i> and <i>Schibilla</i> Raise an SNQ with Respect to Claim 9.....	22
9.	The Combined Teachings of <i>Wolf</i> and <i>Jensen</i> Raise an SNQ with Respect to Claim 11.....	22
10.	Alternatively, the Combined Teachings of <i>Schibilla</i> , <i>Nagar</i> , and <i>Georgiou</i> Raise an SNQ with Respect to Claims 6 and 26	23
11.	Alternatively, the Combined Teachings of <i>Wolf</i> and <i>Georgiou</i> Raise an SNQ with Respect to Claim 6.....	24
12.	Alternatively, the Combined Teachings of <i>Wolf</i> , <i>Christenson</i> , and <i>Georgiou</i> Raise an SNQ with Respect to Claim 26.....	25
VI.	PROPOSED REJECTIONS OF THE '832 PATENT.....	25
VII.	DETAILED EXPLANATION OF THE PROPOSED REJECTIONS.....	26
A.	Ground 1: <i>Schibilla</i> in view of <i>Nagar</i> Renders Obvious Claims 1, 4, 5, 7, 9, 10, 12, 13, 17-20, 22, 23, and 25.....	26
1.	Overview of <i>Schibilla</i>	26
2.	Overview of <i>Nagar</i>	28
3.	Public Availability of <i>Nagar</i>	30
4.	Independent Claim 1	30
a.	[1pre]: “A method for monitoring performance of a storage device”	30

b.	[1a]: “intercepting communications between a computer system and said storage device”	31
c.	[1b]: “analyzing said intercepted communications relative to a threshold value for the performance of said storage device”	36
d.	[1c]: “responding to a decline in the performance of said storage device based on said analyzed intercepted communications by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device”	37
5.	Claim 4	41
6.	Claim 5	43
7.	Claim 7	45
8.	Claim 9	47
9.	Claim 10.....	50
10.	Independent Claim 12	51
a.	[12pre]: “An apparatus for monitoring performance of a storage device”	51
b.	[12a]: “computer readable storage media”	51
c.	[12b]: “computer readable program code stored on said computer readable storage media”	52
d.	[12b(a)]: “program code for intercepting communications between a computer system and said storage device”	52
e.	[12b(b)]: “program code for analyzing said communications, wherein said communications are compared to a predicted decline in performance of said storage device”	52
f.	[12b(c)]: “program code for responding to a decline in the performance of said storage device by backing up at least some data on said storage device prior to said predicted failure of said storage device to enhance continued operation of said storage device”	52
11.	Claim 13.....	53
12.	Claim 17.....	53

13.	Claim 18.....	53
14.	Claim 19.....	54
15.	Claim 20.....	54
16.	Independent Claim 22.....	54
	a. [22pre]: “An apparatus for monitoring performance of a storage device”.....	54
	b. [22a]: “means for evaluating communications between a computer system and said storage device to determine the performance of said storage device based at least in part on intercepted communications with said storage device”.....	55
	c. [22b]: “means for enhancing the performance of said storage device by reallocating at least some data on the storage device”.....	55
17.	Claim 23.....	55
18.	Independent Claim 25.....	56
	a. [25pre]: “A method for monitoring performance of a storage device”.....	56
	b. [25a]: “intercepting communications between a computer system and said storage device”.....	56
	c. [25b]: “analyzing said intercepted communications”.....	56
	a. [25c]: “reallocating at least some of said data on said storage device to enhance the performance of said storage device based on said analyzed communications”.....	56
B.	Ground 2: Claims 2, 3, 6, 14, 15, 24 and 26 Would Have Been Obvious over <i>Schibilla</i> , <i>Nagar</i> , and <i>Hospodor</i>	56
	1. Overview of <i>Hospodor</i>	56
	2. Claim 2.....	57
	3. Claim 3.....	59
	4. Claim 6.....	60
	5. Claim 14.....	61

6.	Claim 15.....	61
7.	Claim 24.....	61
8.	Claim 26.....	62
C.	Ground 3: Claims 8 and 18 Would Have Been Obvious over <i>Schibilla</i> , <i>Nagar</i> , and <i>Hetzler</i>	62
1.	Overview of <i>Hetzler</i>	62
2.	Claim 8	62
3.	Claim 18.....	65
D.	Ground 4: Claims 11 and 16 Would Have Been Obvious over <i>Schibilla</i> , <i>Nagar</i> , and <i>Jensen</i>	65
1.	Overview of <i>Jensen</i>	65
2.	Claim 11.....	66
3.	Claim 16.....	69
E.	Ground 5: Claim 21 Would Have Been Obvious over <i>Schibilla</i> , <i>Nagar</i> , and <i>Voigt</i>	69
1.	Overview of <i>Voigt</i>	69
2.	Claim 21.....	70
F.	Ground 6: Claims 1-4, 6, 7, and 25 Would Have Been Obvious over <i>Wolf</i>	72
1.	Overview of <i>Wolf</i>	72
2.	Independent Claim 1	73
a.	[1pre]: “A method for monitoring performance of a storage device”	73
b.	[1a]: “intercepting communications between a computer system and said storage device”	73
c.	[1b]: “analyzing said intercepted communications relative to a threshold value for the performance of said storage device”	76
d.	[1c]: “responding to a decline in the performance of said storage device based on said analyzed intercepted	

	communications by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device”	79
3.	Claim 2	80
4.	Claim 3	81
5.	Claim 4	82
6.	Claim 6	83
7.	Claim 7	84
8.	Independent Claim 25	85
	a. [25pre]: “A method for monitoring performance of a storage device”	85
	b. [25a]: “intercepting communications between a computer system and said storage device”	85
	c. [25b]: “analyzing said intercepted communications”	85
	d. [25c]: “reallocating at least some of said data on said storage device to enhance the performance of said storage device based on said analyzed communications”	86
G.	Ground 7: Claims 5, 8, 10, and 26 Would Have Been Obvious over <i>Wolf</i> and <i>Christenson</i>	86
	1. Overview of <i>Christenson</i>	86
	2. Claim 5	87
	3. Claim 8	90
	4. Claim 10	94
	5. Claim 26	96
H.	Ground 8: Claim 9 Would Have Been Obvious Over <i>Wolf</i> and <i>Schibilla</i>	97
	1. Claim 9	97
I.	Ground 9: Claim 11 Would Have Been Obvious over <i>Wolf</i> and <i>Jensen</i>	100
	1. Claim 11	100

J.	Ground 10: In the Alternative, Claims 6 and 26 Would Have Been Obvious over <i>Schibilla, Nagar, and Georgiou</i>	102
1.	Overview of <i>Georgiou</i>	102
2.	Claim 6	102
3.	Claim 26.....	104
K.	Ground 11: In the Alternative, Claim 6 Would Have Been Obvious over <i>Wolf</i> and <i>Georgiou</i>	104
1.	Claim 6	105
L.	Ground 12: In the Alternative, Claim 26 Would Have Been Obvious over <i>Wolf, Christenson, and Georgiou</i>	107
1.	Claim 26.....	107
VIII.	CONCLUSION.....	108

I. RELATED PROCEEDINGS

A. Certification Regarding Statutory Estoppel

As required by 37 C.F.R. § 1.510(b)(6), NetApp certifies that the statutory estoppel provisions of 35 U.S.C. § 315(e)(1) and 35 U.S.C. § 325(e)(1) do not prohibit NetApp from filing the instant request for *ex parte* reexamination.

B. Other Proceedings

Patent Owner has asserted the '832 patent in the following district court litigations:

Name	Number	Court	Filed
<i>Valtrus Innovations Ltd. v. Lenovo Grp. Ltd.</i>	2:25-cv-00080	EDTX	Jan. 27, 2025
<i>Valtrus Innovations Ltd. v. Home Depot, Inc.</i>	2:25-cv-00081	EDTX	Jan. 27, 2025
<i>Valtrus Innovations Ltd. v. NetApp Inc.</i>	2:25-cv-00517	EDTX	May 9, 2025
<i>Valtrus Innovations Ltd. v. FedEx Corp.</i>	2:25-cv-00516	EDTX	May 9, 2025

On June 26, 2025, the action against Requester (Ex. L-1) was consolidated with the action against FedEx. *See* Ex. L-2.

On July 25, 2025, Requester filed the following declaratory judgment complaint involving the '832 patent:

Name	Number	Court	Filed
<i>NetApp Inc. v. Valtrus Innovations Ltd.</i>	1-25-cv-06138	SDNY	Jul. 25, 2025

C. Request for Expedited Reexamination

Due to the ongoing nature of the above-identified lawsuit, NetApp respectfully urges that, pursuant to 35 U.S.C. § 305, this Request be granted and reexamination be conducted not only with “special dispatch,” but also with “priority over all other cases” in accordance with M.P.E.P. § 2261.

II. IDENTIFICATION OF CLAIMS THAT SHOULD BE REEXAMINED

Under 37 C.F.R. § 1.510(b)(2), NetApp respectfully requests *ex parte* reexamination of claims 1-26 of the '832 patent and the issuance of a reexamination certificate cancelling these claims.

III. BACKGROUND OF THE TECHNOLOGY

Computer storage devices often face many performance issues as they age. Over time, the performance of storage devices decreases or even fails. Such failures can result in a loss of data. Ex. PA-1, Abstract, 1:8-11, 3:20-48; Ex. C ¶ 40. A common solution to prevent the loss of data is to back up data from one location to another. Ex. PA-1, 8:38-49. As the '832 patent recognizes, the timing of the back-up operation is important to ensure that data is not lost. Ex. A, 1:59-2:21.

There are several ways to implement timing of a back-up operation. One option is to back-up the data on a regular time interval. *Id.*, 2:8-10. Another option involves monitoring the performance of a storage device and backing up the stored data once a specific performance threshold is reached. *See* Ex. PA-1, Abstract, 4:15-41, 6:45-7:10, Figs. 3-5. This operation may occur automatically when the specific performance threshold is reached. Ex. PA-1, 6:62-7:10, 8:30-49.

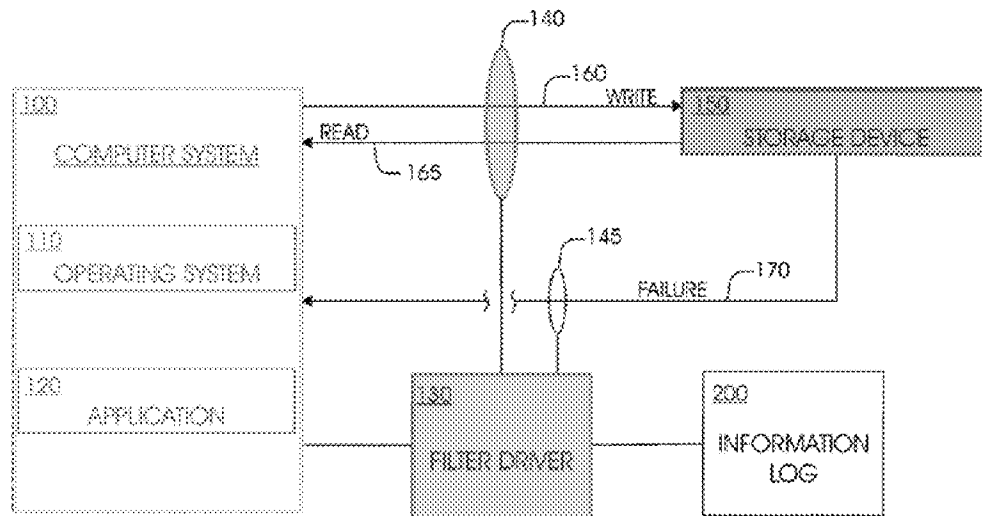
In addition, a storage device may be defragmented in response to a decline in performance. The defragmentation process generally comprises identifying fragmented files, determining whether the disk contains sufficient free space, moving some of or the entire file to contiguous locations, and consolidating free space. *See* Ex. PA-5, 3:49-4:4, Figs. 4-9; *see also*, *e.g.*, *id.*, 8:24-9:30 (describing a partial and total defragmentation), 9:33-10:8 (describing moving files), 13:65-14:47. This process minimizes file fragmentation and the number of accesses required to transfer files. *Id.*, 3:5-11.

These solutions, including monitoring the storage device, backing up data, and performing file fragmentation, all contribute to enhanced performance and operation of the storage device. Ex. PA-1, 7:1-10; Ex. PA-5, 2:41-48. Each of these concepts was well understood prior to the time the application leading to the '832 patent was filed.

IV. OVERVIEW OF THE '832 PATENT AND ITS PROSECUTION HISTORY

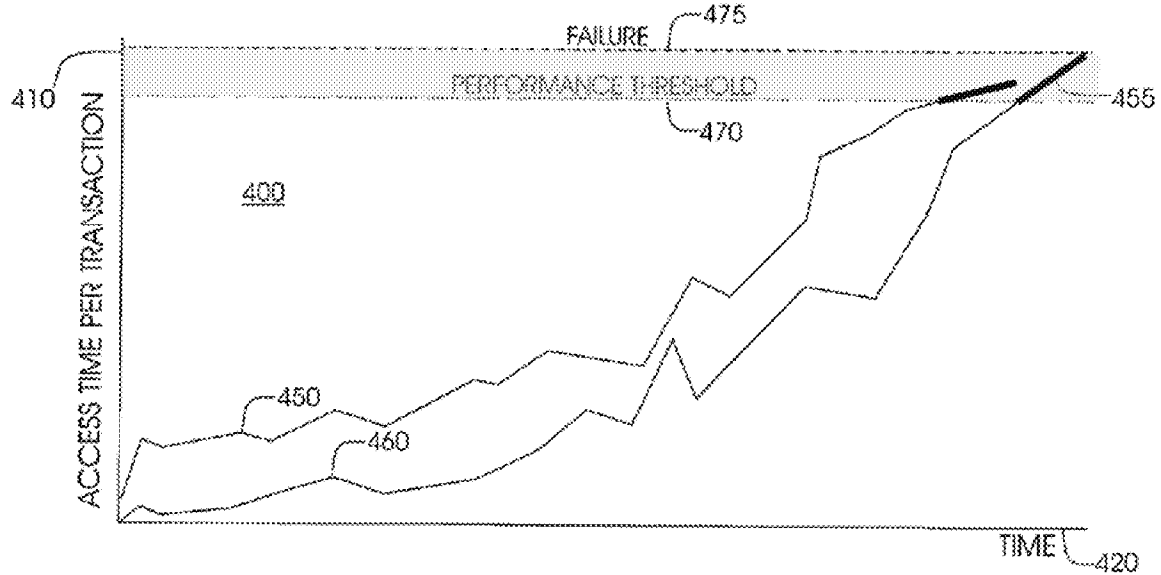
A. Specification

The '832 patent relates to monitoring performance of a storage device. *See, e.g.*, Ex. A, Abstract. The '832 system intercepts communications between a storage device and a computer system, such as read and write commands (160, 165 below), using software such as a filter driver (130):

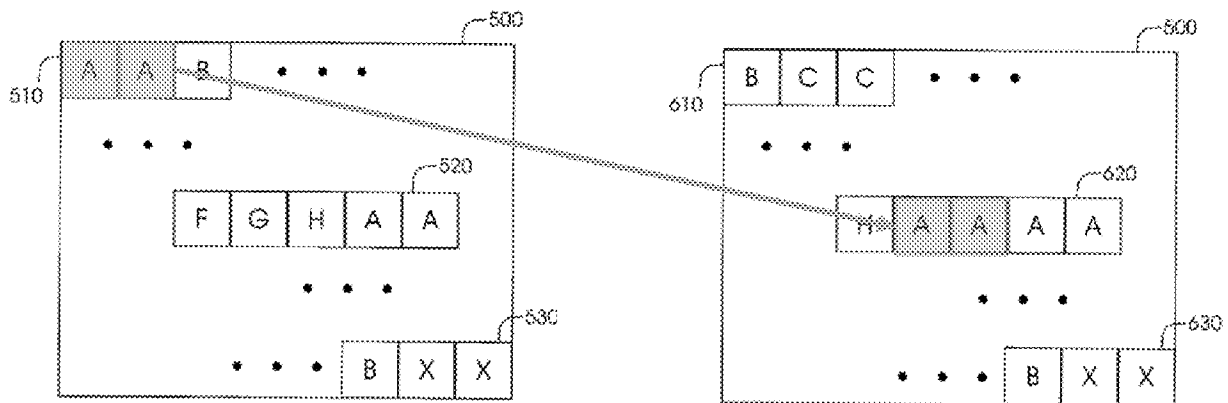


Id., Fig. 1.

The system monitors these communications and determines a performance metric for the storage device. *See id.*, 6:16-36. These metrics may include error rates or the time required to access data. *See id.*, 6:16-36, 7:7-13, Figs. 3-4. The system then compares this metric to a performance threshold:



Id., Figs. 3-4. When the device's performance crosses this threshold, the system backs up or reallocates the data to an alternative location within the same device to prevent data loss due to potential device failure:



See id., Figs. 5-6, 6:16-36, 8:8-37, 10:25-49.

B. Claims 1-26 of the '832 Patent

This Request challenges claims 1-26, reproduced below. Ex. A, 11:34-14:15. Dependent claims 2-11 depend from independent claim 1, dependent claims 13-21 depend from independent claim 12, and dependent claims 23-26 depend from independent claim 22.

1. A method for monitoring performance of a storage device, comprising:

intercepting communications between a computer system and said storage device;

analyzing said intercepted communications relative to a threshold value for the performance of said storage device; and

responding to a decline in the performance of said storage device based on said analyzed intercepted communications by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device.

2. A method as in claim 1, further comprising measuring access time for said storage device.

3. A method as in claim 2, further comprising correcting said measured access time for system overhead.

4. A method as in claim 1, wherein intercepting said communications comprises intercepting an error reported by said storage device.

5. A method as in claim 1, further comprising determining an access location on said storage device and an access frequency for data stored thereon, based on said intercepted communications.

6. A method as in claim 1, further comprising determining an access location on said storage device and an access duration for data stored thereon, based on said intercepted communications.

7. A method as in claim 1, further comprising logging said communications over time.

8. A method as in claim 7, wherein analyzing said communications comprises deriving said threshold value based on said logged communications.

9. A method as in claim 1, wherein responding to said declining performance of said storage device further comprises automatically backing-up data stored on said storage device.

10. A method as in claim 1, wherein reallocating at least some data on said storage device is based on usage patterns of said data.

11. A method as in claim 1, wherein responding to said declining performance of said storage device further comprises defragmenting at least a portion of said storage device.

12. An apparatus for monitoring performance of a storage device, comprising:

computer readable storage media;

computer readable program code stored on said computer readable storage media, comprising:

a) program code for intercepting communications between a computer system and said storage device;

b) program code for analyzing said communications, wherein said communications are compared to a predicted decline in performance of said storage device; and

c) program code for responding to a decline in the performance of said storage device by backing up at least some data on said storage device prior to said predicted failure of said storage device to enhance continued operation of said storage device.

13. An apparatus as in claim 12, wherein said program code for intercepting said communications comprises program code for intercepting an error reported by said storage device.

14. An apparatus as in claim 12, further comprising program code for measuring access time for said storage device, and wherein said communications comprises at least said access time.

15. An apparatus as in claim 14, further comprising:

a) program code for determining system overhead; and

b) program code for correcting said access time for said system overhead.

16. An apparatus as in claim 12, wherein said program code for responding to said decline in the performance of said storage device further comprises defragmenting at least a portion of said storage device based on said communications.

17. An apparatus as in claim 12, further comprising:

a communications log;

program code for logging said communications over time in said communications log; and

program code for determining the decline in performance of said storage device based at least in part on said logged communications.

18. An apparatus as in claim 12, further comprising program code for deriving a threshold value for the performance of said storage device, wherein said program code for responding to said decline in the performance of said storage device responds when the performance of said storage device satisfies said threshold value thereof.

19. An apparatus as in claim 12, wherein said program code for responding further comprises program code for reallocating data to another sector of said storage device.

20. An apparatus as in claim 19, wherein said program code for reallocating reallocates at least some data on said storage device based on usage patterns of said at least some data.

21. An apparatus as in claim 12, further comprising a graphical user interface for reporting the performance of said storage device to a user.

22. An apparatus for monitoring performance of a storage device, comprising:

means for evaluating communications between a computer system and said storage device to determine the performance of said storage device based at least in part on intercepted communications with said storage device; and

means for enhancing the performance of said storage device by reallocating at least some data on the storage device.

23. An apparatus as in claim 22, further comprising means for intercepting communications with said storage device.

24. An apparatus as in claim 22, further comprising means for measuring time to access said storage device, wherein said measured access time is evaluated to determine the performance of said storage device.

25. A method for monitoring performance of a storage device, comprising:

intercepting communications between a computer system and said storage device;

analyzing said intercepted communications; and

reallocating at least some of said data on said storage device to enhance the performance of said storage device based on said analyzed communications.

26. A method as in claim 25, wherein intercepting said communications comprises determining access location, access frequency, and access duration for said data on said storage device.

Id., 11:34-14:10.

C. Prosecution History

The application leading to the '832 patent (U.S. Application No. 09/966,953) was filed September 27, 2001. Ex. A, Cover Page, (21)-(22) The Examiner initially rejected the pending claims as obvious over various references. Ex. B at 49-56. In response, Applicant amended the pending claims to recite “automatically reallocating” or “backing up” “at least some data on said storage device.” *Id.* at 67, 69, 72; *see also id.* at 74-79 (arguing the prior art failed to teach these features). The Examiner issued a Final Office Action, finding these limitations taught in other art of record, namely, Archibald. *See id.* at 88-96.

Applicant traversed the rejection without amendment. To do so, the Applicant submitted a Rule 131 affidavit and other evidence to swear behind Archibald (dated June 1, 2001). *See id.* at 109-130. The Office initially rejected the affidavit, finding it insufficient to establish earlier

conception. *Id.* at 139-141. Applicant submitted an RCE, urged that its evidence was sufficient, and submitted new evidence for further consideration. *See id.* at 154-200.

The Office accepted the affidavit on conception but still rejected the claims as indefinite and obvious over new art. *Id.* at 215-225. Applicant amended the claims to recite the invention “enhance[s] continued operation of the storage device.” *Id.* at 232, 234, 237 (similar). According to Applicant, removing a failed disk drive from an array—as taught in the art—did not teach this feature. *Id.* at 240. The Examiner maintained the rejection in a Final Office Action. *Id.* at 257-270.

Applicant appealed. *Id.* at 274. Applicant argued so-called S.M.A.R.T. drives cited by the Examiner did not “intercept[] communications” between a disk drive and a host computer. *See id.* at 282-283. It also argued the claims did not attempt to cover reallocating data in “an array” of storage disks. *Id.* at 283-284. Instead, it contended the claims covered reallocating data to enhance continued operation of the same (supposedly singular) storage device. *Id.* The Examiner thereafter issued a Notice of Allowance, finding the pending claims patentable. *Id.* at 318.

D. Level of Ordinary Skill in the Art

A person of ordinary skill in the art (“POSITA”) at the time the application leading to the ’832 patent was filed would have had a bachelor’s degree in electrical engineering, computer science, or a similar field and three years of experience designing or analyzing disk drive hardware, firmware, or software. Ex. C ¶ 52. Additional relevant experience could substitute for formal education, and additional relevant education could substitute for work experience. *Id.*

E. Claim Construction for the ’832 Patent

The ’832 patent has expired, and thus the Office should construe the claims under the principles of *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005) (en banc). *See* MPEP § 2258 I.G.

NetApp presents certain constructions for terms invoking pre-AIA 35 U.S.C. § 112, ¶ 6. These constructions include the noted functions and corresponding structure—all being simple, conventional components—as well as equivalents of such structure. NetApp does not concede the '832 patent discloses adequate structure, such as algorithms, for purposes of definiteness for any claim term.¹

1. “means for evaluating” (claim 22)

Function: evaluating communications between a computer system and a storage device.

Ex. A, claim 22; Ex. C ¶ 47.

Structure: Hardware, “firmware, and/or software (i.e., computer readable program code), stored in computer readable storage media and executed, for example, by [a] computer system” comprising “any conventional computer, such as a desktop personal computer (PC), laptop PC, network workstation, network server, etc.,” ostensibly by a processor. Ex. A, 3:59-4:4, 4:17-31; Ex. C ¶ 47.

2. “means for enhancing” (claim 22)

Function: enhancing the performance of a storage device by reallocating at least some data on the storage device. Ex. A, claim 22; Ex. C ¶ 48.

Structure: Hardware, “firmware, and/or software (i.e., computer readable program code), stored in computer readable storage media and executed, for example, by [a] computer system” comprising “any conventional computer, such as a desktop personal computer (PC), laptop PC, network workstation, network server, etc.,” ostensibly by a processor. Ex. A, 3:59-4:4, 4:17-31; Ex. C ¶ 48.

¹ Requester reserves the right to argue claim construction in other fora and respond in this forum to any construction Patent Owner may advance. *See, e.g.*, 37 C.F.R. § 1.535.

3. “means for intercepting” (claim 23)

Function: intercepting communications with a storage device. Ex. A, claim 23; Ex. C ¶ 49.

Structure: program code comprising or operating with a filter driver. *See* Ex. A, 3:47-58 (describing “a filter driver 130 for intercepting 140 communications (e.g., write commands 160, read commands 165) between [a] computer system 100 (e.g., the operating system 110, and application 120) and [a] storage device 150”), 4:4-16 (“The program code for intercepting the communications may comprise, or operate in conjunction with, [a] filter driver.”), Fig. 1; Ex. C ¶ 49.

4. “means for measuring” (claim 24)

Function: measuring time to access a storage device. Ex. A, claim 24; Ex. C ¶ 50.

Structure: Hardware, “firmware, and/or software (i.e., computer readable program code), stored in computer readable storage media and executed, for example, by [a] computer system” comprising “any conventional computer, such as a desktop personal computer (PC), laptop PC, network workstation, network server, etc.,” ostensibly by a processor. *See* Ex. A, 7:7-12 (explaining how “the program code for analyzing the intercepted communications may find lengthening execution or access times for the storage device”); *see also id.*, 3:59-4:4, 4:17-31; Ex. C ¶ 50.

V. IDENTIFICATION OF PRIOR ART AND SUBSTANTIAL NEW QUESTIONS OF PATENTABILITY

A. Identification of Prior Art

The earliest possible priority date for claims 1-26 of the '832 patent is September 27, 2001. Ex. A, (22). Reexamination is requested in view of the following prior art references:

PA-1: U.S. Patent No. 6,384,999 to Mark Schibilla, filed April 12, 1999 (“*Schibilla*”)

- PA-2: Rajeev Nagar, *Windows NT File System Internals* (O'Reilly & Associates, Inc., 1st ed. September 1997) (“*Nagar*”)
PA-3: U.S. Patent No. 4,851,998 to Andrew Hospodor, filed June 3, 1987 (“*Hospodor*”)
PA-4: U.S. Patent No. 5,682,273 to Stephen Hetzler, filed September 22, 1995 (“*Hetzler*”)
PA-5: U.S. Patent No. 5,930,828 to Craig Jensen et al., filed March 26, 1997 (“*Jensen*”)
PA-6: U.S. Patent No. 5,623,598 to Douglas Voigt et al., filed November 22, 1994 (“*Voigt*”)
PA-14: U.S. Patent No. 5,287,363 to Paul I. Wolf et al., filed July 1, 1991 (“*Wolf*”)
PA-15: U.S. Patent No. 6,324,620 to Patrick J. Christenson et al., filed July 23, 1998 (“*Christenson*”)
PA 16: U.S. Patent No. 5,774,292 to Christos J. Georgiou et al., filed April 13, 1995 (“*Georgiou*”)

These prior art documents are listed on a form PTO/SB/08 filed concurrently herewith. Copies of all the listed references are attached.

B. Substantial New Questions of Patentability

For an SNQ to be present, “it is only necessary that: (A) an item of information raises a[n] SNQ regarding at least one claim, i.e., the teaching of the item of information is such that a reasonable examiner would consider the teaching to be important in deciding whether or not the claim is patentable; and (B) the same question of patentability . . . has not been decided by the Office in an earlier concluded examination or review of the patent, decided in a final holding of invalidity (after all appeals) by a federal court in a decision on the merits involving the claim, or raised to or by the Office in a pending reexamination or supplemental examination of the patent.” M.P.E.P. § 2816.02(I); *see also In re Vivint, Inc.*, 14 F.4th 1342, 1349 (Fed. Cir. 2021). “For example, the same question of patentability may already have been decided if the Office has previously considered, in an earlier concluded examination or review of the patent, the same question of patentability based on the same prior art.” *See* M.P.E.P. § 2816.02(I) (citing *In re Recreative Techs. Corp.*, 83 F.3d 1394 (Fed. Cir. 1996)).

Each proposed rejection in this Request presents a distinct SNQ of patentability. First, a reasonable Examiner would find the teachings of the prior art submitted in this Request (as also

discussed in detail below) important in deciding whether the claims are patentable. Second, the same question of patentability has not been decided by the Office in an earlier concluded review or examination of the '832 patent. During prosecution, the Examiner did not consider any of the prior art relied upon in this Request. *See, e.g.*, Section V.B.1-12. Third, the same question of patentability has not been decided in a final holding of invalidity by a federal court. Though the '832 patent is currently involved in co-pending district court litigations, those litigations have not resolved any question of validity regarding the '832 claims. *See supra* Section I.B. And finally, the same question of patentability has not been raised to or by the Office in a pending reexamination or supplemental examination. To NetApp's best knowledge, there has been no previous Office proceedings, *inter partes* review or *ex parte* reexamination, involving the claims of the '832 patent, much less any involving the specific combination of prior art that NetApp has raised below.

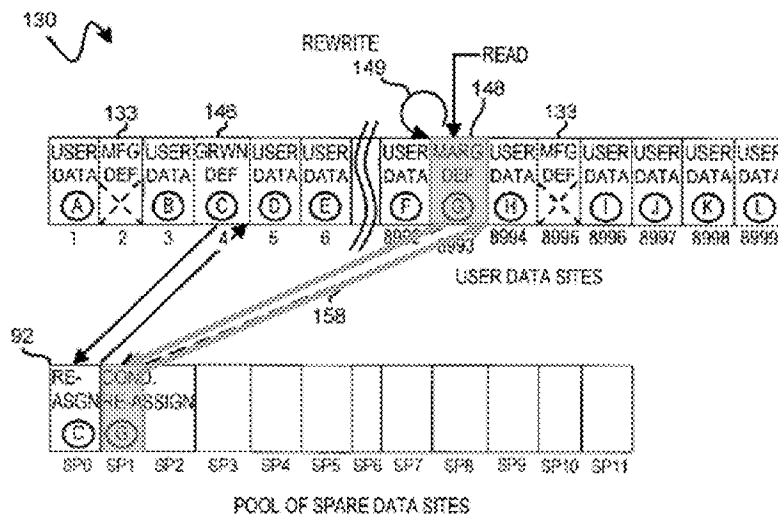
1. The Combined Teachings of *Schibilla* and *Nagar* Raise an SNQ with Respect to Claims 1, 4, 5, 7, 9, 10, 12, 13, 17-20, 22, 23, and 25

Independent claims 1 and 12 of the '832 patent are directed to a method and apparatus for “*monitoring performance of a storage device.*”² Ex. A, 11:34-46; 12:12-27. The claimed method and apparatus include, in part, “*responding to a decline in the performance of said storage device by automatically reallocating*” or “*backing up*” “*some of data on said storage device.*” *Id.*, 11:41-46; 12:23-27. Independent claims 22 and 25 have similar limitations. *Id.*, 13:4-13; 14:3-10. Claims 4, 5, 7, 9, 10, 13, 17-20, and 23 depend from one of independent claims 1, 12, 22, or 25, directly or indirectly, and thus incorporate the limitations of the independent claims.

² For clarity, this request hereinafter italicizes claim language of the '832 patent to better distinguish it from quotations from other sources (e.g., the prior art).

During prosecution, the Examiner found that the prior art of record does not teach or suggest these limitations. But these claimed concepts were known before the earliest claimed priority date of the '832 patent, as demonstrated by *Schibilla* and *Nagar*.

Schibilla discloses “*responding*” to a drop in performance “*by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device.*” See Ex. PA-1, 6:62-7:10. For example, when *Schibilla* encounters a marginally defective data site (defined by its scoring algorithm), *Schibilla* “recovers data from the marginally defective data site [by] writ[ing] the recovered data to a spare data site elsewhere on the disk surface.” *Id.* This involves “writing the data from the marginally defective data site to an alternative location” to “provide[] a backup copy of the data,” as shown below in Figure 5:



Id., 6:62-7:10, Fig. 5. *Schibilla* also analyzes communications against a “*threshold*” by using a “scoring algorithm” to compare data requests to a “predefined performance threshold.” *Id.*, Abstract, 4:15-41, 6:31-7:10, 8:5-29, Figs. 3-5.

Nagar also provides additional disclosure regarding “*intercepting communications between a computer system and said storage device*” using a filter driver. See Ex. PA-2, 22, 23.

Nagar describes how to implement a “filter driver” in software that “intercepts all I/O requests targeted to a specified mounted logical volume.” *Id.*, xvii. These filter drivers “intercept[] requests targeted to some existing software module,” including a “file system” or “disk driver” exchanging information with a disk (“*intercept[] communications between a computer system and said storage device*”). *Id.*, 22, 33. This enables the driver to “extend, or simply replace, the functionality provided by the original recipient of the request” via software. *Id.*, 33-34.

Schibilla and *Nagar* were not previously considered or addressed during the prosecution of the application resulting in the ’832 patent. Neither were cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Because *Schibilla* and *Nagar* teach features that the Examiner did not previously find in the prior art, and the Examiner did not consider these references, the teachings of *Schibilla* and *Nagar* raise an SNQ for the patentability of claims 1, 4, 5, 7, 9, 10, 12, 13, 17-20, 22, 23, and 25.

2. The Combined Teachings of *Schibilla*, *Nagar*, and *Hospodor* Raise an SNQ with Respect to Claims 2, 3, 6, 14, 15, 24, and 26

Claims 2, 3, 6, 14, 15, 24, and 26 depend from one of independent claims 1, 12, 22, or 25, either directly or indirectly, and thus incorporate the limitations of the independent claims. As stated in Section V.B.1, the teachings of *Schibilla* and *Nagar* raise an SNQ with respect to claims 1, 12, 22, and 25.

The combined teachings of *Schibilla*, *Nagar*, and *Hospodor* raise an SNQ with respect to claims 2, 3, 6, 14, 15, 24, and 26. For example, *Hospodor* provides additional detailed disclosures regarding “*measuring access time for said storage device*” and “*correcting said measured access time for system overhead*.” See Ex. PA-3, 3:64-4:4. *Hospodor* describes techniques for measuring “performance characteristics” of a disk drive, including “access time” (“*measuring access time*”). See, e.g., *id.*, Abstract, 1:7-26, 1:59-2:34, 5:25-42, 8:13-48.

Hospodor further describes subtracting a “host overhead” time (t_{HO} or $t_{HOST\ OVERHEAD}$) from “the total amount of time necessary to completely carry out a particular instruction” (t_{TOTAL}). Ex. PA-3, 4:22-54, 5:5-42, 7:43-63, 8:13-48. In certain aspects, *Hospodor* describes this host overhead as accounting for delays attributable to “the host system,” such a CPU’s clock frequency and execution efficiency, rather than the storage device itself. *See, e.g.*, Ex. PA-3, 2:9-44, 4:22-54, 5:5-42, 8:31-48.

Hospodor was not previously considered or addressed during the prosecution of the application resulting in the ’832 patent. *Hospodor* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Hospodor*, in combination with *Schibilla* and *Nagar*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Schibilla*, *Nagar*, and *Hospodor* raise an SNQ for the patentability of claims 2, 3, 6, 14, 15, 24, and 26.

3. The Combined Teachings of *Schibilla*, *Nagar*, and *Hetzler* Raise an SNQ with Respect to Claims 8 and 18

Claims 8 and 18 depend from one of the independent claims 1 or 12, either directly or indirectly, and thus incorporate the limitations of those independent claims. As stated in Section V.B.1, the teachings of *Schibilla* and *Nagar* raise an SNQ with respect to claims 1 and 12.

The combined teachings of *Schibilla*, *Nagar*, and *Hetzler* raise an SNQ with respect to claims 8 and 18. For example, *Hetzler* provides additional detailed disclosures regarding analyzing communications and “*deriving said threshold value based on said logged communications.*” *See* Ex. PA-4, 2:16-63. *Hetzler* monitors a disk drive’s workload by analyzing the read and write operations accessing the disk. *See id.*, Abstract, 2:16-63. In some embodiments, *Hetzler* improves the disk’s energy performance by placing the disk into a power-saving mode depending on these access patterns. *See id.*, Abstract, 2:16-63. *Hetzler* selects the

appropriate mode by comparing current access demands to a “previously calculated and continually updated threshold.” *Id.*, 2:16-63. The system updates these thresholds by adjusting factors “in response to tracking the actual performance of the system.” Ex. PA-4, 2:16-63. These dynamic thresholds allow *Hetzler* to “adapt dynamically to varying workload situations.” *Id.*, 2:16-63.

Hetzler was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Hetzler* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Hetzler*, in combination with *Schibilla* and *Nagar*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Schibilla*, *Nagar*, and *Hetzler* raise an SNQ for the patentability of claims 8 and 18.

4. The Combined Teachings of *Schibilla*, *Nagar*, and *Jensen* Raise an SNQ with Respect to Claims 11 and 16

Claims 11 and 16 depend from one of independent claims 1 or 12 and thus incorporate the limitations of those independent claims. As stated in Section V.B.1, the teachings of *Schibilla* and *Nagar* raise an SNQ with respect to claims 1 and 12.

The combined teachings of *Schibilla*, *Nagar*, and *Jensen* raise an SNQ with respect to claims 11 and 16. For example, *Jensen* provides additional detailed disclosures regarding “defragmenting at least a portion of said storage device.” See Ex. PA-5, 8:24-9:30. *Jensen* describes techniques for defragmenting disk drives in a computer system, particularly those running Windows NT. See *id.*, Abstract, 1:6-9, 2:32-48, 3:49-61. *Jensen*’s defragmentation process generally comprises identifying fragmented files, determining whether the disk contains sufficient free space, moving some of or the entire file to contiguous locations, and consolidating

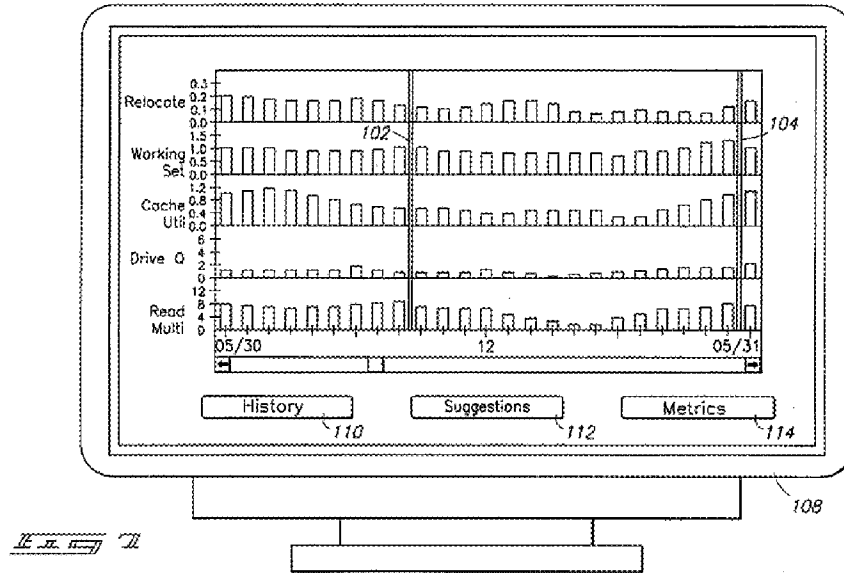
free space. *See* Ex. PA-5, 3:49-4:4, Figs. 4-9; *see also, e.g., id.*, 8:24-9:30 (describing a partial and total defragmentation), 9:33-10:8 (describing moving files), 13:65-14:47.

Jensen was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Jensen* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Jensen*, in combination with *Schibilla* and *Nagar*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Schibilla*, *Nagar*, and *Jensen* raise an SNQ for the patentability of claims 11 and 16.

5. The Combined Teachings of *Schibilla*, *Nagar*, and *Voigt* Raise an SNQ with Respect to Claim 21

Claim 21 depends from independent claim 12 and thus incorporates the limitations of those independent claims. As stated in Section V.B.1, the teachings of *Schibilla* and *Nagar* raise an SNQ with respect to claim 12.

The combined teachings of *Schibilla*, *Nagar*, and *Voigt* raise an SNQ with respect to claim 21. For example, *Voigt* provides additional detailed disclosures regarding “a graphical user interface for reporting the performance of said storage device to a user.” *See* Ex. PA-6, 6:10-36. *Voigt* measures various “performance metrics,” such as blocks read, wait queues, and access times. *Id.*, 4:64-5:13, 5:39-65. *Voigt* can display this information to the user via a graphical user interface:



See Ex. PA-6, 6:10-36 (“Preferably, the performance history is graphically displayed on visual display screen 18 in Fig. 1.”), 8:25-34, Fig. 7, claims 4-5, 11, 15.

Voigt was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Voigt* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Voigt*, in combination with *Schibilla* and *Nagar*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Schibilla*, *Nagar*, and *Voigt* raise an SNQ for the patentability of claim 21.

6. The Teachings of *Wolf* Raise an SNQ with Respect to Claims 1-4, 6, 7, and 25

As explained in Section V.B.1, independent claim 1 of the '832 patent is directed to a method for “*monitoring performance of a storage device.*” Ex. A, 11:34-46; 12:12-27, 14:3-10. The claimed method includes, in part, “*responding to a decline in the performance of said storage device by automatically reallocating*” or “*backing up*” “*some of data on said storage device.*” *Id.*, 11:41-46; 12:23-27. Independent claim 25 has similar limitations. *Id.*, 14:3-10.

Claims 2-4, 6, and 7 depend from independent claim 1, directly or indirectly, and thus incorporate the limitations of independent claim 1.

During prosecution, the Examiner found that the prior art of record does not teach or suggest these limitations. But these claimed concepts were known before the earliest claimed priority date of the '832 patent, as demonstrated by *Wolf*.

Wolf discloses a “disk call interceptor” (DCI) for intercepting requests to retrieve data from a storage device. PA-14, 21:24-43. *Wolf* reviews these intercepted communications to monitor system performance, such as identifying read problems, computing data transfer times, and logging this information to a file. *Id.*, 6:57-7:11. When the system detects transfers exceeding a given amount of time (i.e., taking “an excessive amount of time”), it relocates data to a different location on disk. *Id.*; *see also, e.g., id.*, Abstract; 3:40-53; 15:41-65; 17:4-37.

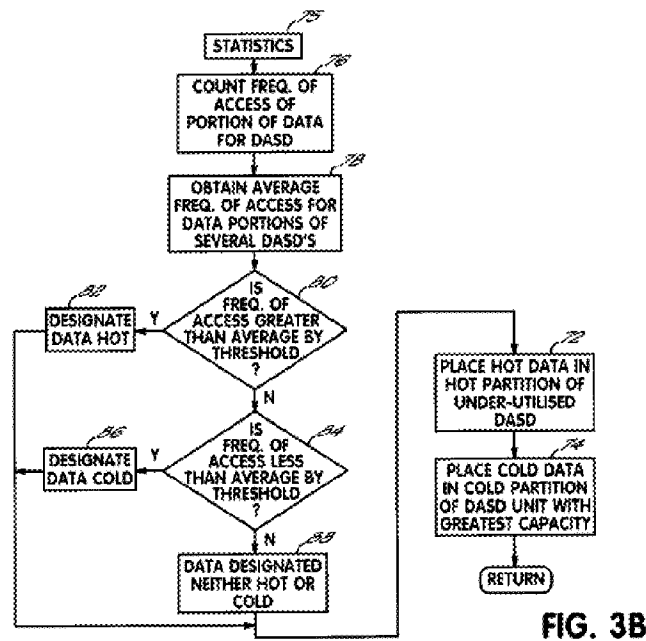
Wolf was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Wolf* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Because *Wolf* teaches features that the Examiner did not previously find in the prior art, and the Examiner did not consider these references, the teachings of *Wolf* raise an SNQ for the patentability of claims 1-4, 6, 7, and 25.

7. The Combined Teachings of *Wolf* and *Christenson* Raise an SNQ with Respect to Claims 5, 8, 10, and 26

Claims 5, 8, 10, and 26 depend from one of the independent claims 1 or 25, either directly or indirectly, and thus incorporate the limitations of those independent claims. As explained in Section V.B.6, the teachings of *Wolf* raise an SNQ with respect to claims 1 and 25.

The combined teachings of *Wolf* and *Christenson* raise an SNQ with respect to claims 5, 8, 10, and 26. For example, *Christenson* provides additional detailed disclosures regarding, for

example, “determining an access location on said storage device and an access frequency for data stored thereon.” See PA-15, Abstract, 2:51-62, 8:12-9:20, Fig. 3B. *Christenson* measures the “access frequency of data” on disk and designates the data as “HOT” (frequently accessed), COLD (infrequently accessed), or neither (average access) based in part on these frequencies:



See *id.*, 2:51-62, 8:12-9:20, Fig. 3B. *Christenson* then reallocates data to HOT and COLD partitions of a disk (or among multiple disks) based on these designations and other considerations (including, for example, how often the disk itself is used). See, e.g., *id.*, 2:63-3:20, 6:46-7:10, 9:26-54, 10:56-11:16, Figs. 5-6.

Christenson was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Christenson* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Christenson*, in combination with *Wolf*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Wolf* and *Christenson* raise an SNQ for the patentability of claims 5, 8, 10, and 26.

8. The Combined Teachings of *Wolf* and *Schibilla* Raise an SNQ with Respect to Claim 9

Claim 9 depends from independent claim 1, either directly or indirectly, and thus incorporates its limitations. As explained in Section V.B.6, the teachings of *Wolf* raise an SNQ with respect to claim 1.

The combined teachings of *Wolf* and *Schibilla* raise an SNQ with respect to claim 9. For example, *Schibilla* provides additional detailed disclosures regarding “*responding to said declining performance of said storage device further comprises automatically backing-up data stored on said storage device.*” See PA-1, 6:62-7:10, 8:30-49, 10:15-41. *Schibilla* explains how its automatic conditional reassignment creates a “backup copy of the data” on the disk (“*automatically backing-up data stored on said storage device*”). *Id.* And this conditional reassignment occurs when a disk’s performance breaches a threshold (“*responding to said declining performance of said storage device*”). See, e.g., Abstract, 4:15-41, 6:45-7:10. As discussed, *Schibilla* was not previously considered or addressed during the prosecution of the application resulting in the ’832 patent. *Schibilla* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Schibilla*, in combination with *Wolf*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Wolf* and *Schibilla* raise an SNQ for the patentability of claim 9.

9. The Combined Teachings of *Wolf* and *Jensen* Raise an SNQ with Respect to Claim 11

Claim 11 depends from independent claim 1 and thus incorporates its limitations. As stated in Section V.B.6, the teachings of *Wolf* raise an SNQ with respect to claims 1.

The combined teachings of *Wolf* and *Jensen* raise an SNQ with respect to claim 11. For example, *Jensen* provides additional detailed disclosures regarding “*defragmenting at least a*

portion of said storage device.” See Ex. PA-5, 8:24-9:30. Jensen describes techniques for defragmenting disk drives in a computer system, particularly those running Windows NT. See id., Abstract, 1:6-9, 2:32-48, 3:49-61. Jensen’s defragmentation process generally comprises identifying fragmented files, determining whether the disk contains sufficient free space, moving some of or the entire file to contiguous locations, and consolidating free space. See Ex. PA-5, 3:49-4:4, Figs. 4-9; see also, e.g., id., 8:24-9:30 (describing a partial and total defragmentation), 9:33-10:8 (describing moving files), 13:65-14:47.

As discussed, *Jensen* was not previously considered or addressed during the prosecution of the application resulting in the ’832 patent. *Jensen* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Jensen*, in combination with *Wolf*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Wolf* and *Jensen* raise an SNQ for the patentability of claim 11.

10. Alternatively, the Combined Teachings of *Schibilla*, *Nagar*, and *Georgiou* Raise an SNQ with Respect to Claims 6 and 26

Claims 6 and 26 depend from one of the independent claims 1 or 25, either directly or indirectly, and thus incorporate the limitations of those independent claims. As explained in Section V.B.1, the teachings of *Schibilla* and *Nagar* raise an SNQ with respect to claims 1 and 25.

The combined teachings of *Schibilla*, *Nagar*, and *Georgiou* raise an SNQ with respect to claim 6. For example, *Georgiou* provides additional detailed disclosures regarding “determining ... an access duration of data stored thereon.” *Georgiou* describes ways for optimizing power consumption used by a disk drive. See PA-16, Abstract. *Georgiou* uses control logic to determine, access times, the “duration of current disk activity,” and the “duration” of a

disk driver's operational state. *Id.*, 2:56-3:6. *Georgiou* uses logs histories of disk activity and inactivity to predict future periods of activity and inactivity for the drive. *See, e.g., id.*, 2:36-46. Using these predictive algorithms, *Georgiou* alters the drive's rotational speed to improve a disk's power consumption profile. *Id.* In this way, *Georgiou* measures the access duration of data stored on disk.

Georgiou was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Georgiou* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Georgiou*, in combination with *Schibilla* and *Nagar*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Schibilla*, *Nagar*, and *Georgiou* raise an SNQ for the patentability of claims 6 and 26.

11. Alternatively, the Combined Teachings of *Wolf* and *Georgiou* Raise an SNQ with Respect to Claim 6

Claim 6 depends from independent claims 1 and thus incorporates its limitations. As explained in Section V.B.6, the teachings of *Wolf* raise an SNQ with respect to claims 1.

The combined teachings of *Wolf* and *Georgiou* raise an SNQ with respect to claim 6. As discussed above, for example, *Georgiou* teaches determining an “access duration.” *See* § V.B.10.

As discussed, *Georgiou* was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Georgiou* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Georgiou*, in combination with *Wolf*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Wolf* and *Georgiou* raise an SNQ for the patentability of claim 6.

12. Alternatively, the Combined Teachings of *Wolf*, *Christenson*, and *Georgiou* Raise an SNQ with Respect to Claim 26

Claim 26 depends from independent claim 25 and thus incorporates its limitations. As explained in Section V.B.7, the teachings of *Wolf* and *Christenson* raise an SNQ with respect to claim 26.

The combined teachings of *Wolf*, *Christenson*, and *Georgiou* raise an SNQ with respect to claim 26. As discussed above, *Christenson* teaches determining an “*access frequency*” (see § V.B.7), and *Georgiou* teaches determining an “*access duration*” (see § V.B.10).

As discussed, *Georgiou* was not previously considered or addressed during the prosecution of the application resulting in the '832 patent. *Georgiou* was not cited on any information disclosure statement or considered by the Examiner at any point during prosecution. Ex. A, (56). Therefore, because *Georgiou*, in combination with *Wolf* and *Christenson*, further teaches claim features the Examiner thought were missing from the prior art, the teachings of *Wolf*, *Christenson*, and *Georgiou* raise an SNQ for the patentability of claim 6.

VI. PROPOSED REJECTIONS OF THE '832 PATENT

The following rejections are proposed in this Request for 1-26 of the '832 patent:

- Claims 1, 4, 5, 7, 9, 10, 12, 13, 17-20, 22, 23, and 25 are unpatentable under 35 U.S.C. § 103 as being obvious over *Schibilla* in view of *Nagar*.
- Claims 2, 3, 6, 14, 15, 24, and 26 are unpatentable under 35 U.S.C. § 103 as being obvious over *Schibilla* in view of *Nagar* and *Hospodor*.
- Claims 8 and 18 are unpatentable under 35 U.S.C. § 103 as being obvious over *Schibilla* in view of *Nagar* and *Hetzler*.
- Claims 11 and 16 are unpatentable under 35 U.S.C. § 103 as being obvious over *Schibilla* in view of *Nagar* and *Jensen*.

- Claim 21 is unpatentable under 35 U.S.C. § 103 as being obvious over *Schibilla* in view of *Nagar* and *Voigt*.
- Claims 1-4, 6, 7, and 25 are unpatentable under 35 U.S.C. § 103 as being obvious over *Wolf*.
- Claims 5, 8, 10, and 26 are unpatentable under 35 U.S.C. § 103 as being obvious over *Wolf* in view of *Christenson*.
- Claim 11 is unpatentable under 35 U.S.C. § 103 as being obvious over *Wolf* in view of *Jensen*.
- Claims 6 and 26 are unpatentable under 35 U.S.C. § 103 as being obvious over *Schibilla* in view of *Nagar* and *Georgiou*.
- Claim 6 is unpatentable under 35 U.S.C. § 103 as being obvious over *Wolf* in view of *Georgiou*.
- Claim 26 is unpatentable under 35 U.S.C. § 103 as being obvious over *Wolf* in view of *Christenson* and *Georgiou*.

VII. DETAILED EXPLANATION OF THE PROPOSED REJECTIONS

A. Ground 1: *Schibilla* in view of *Nagar* Renders Obvious Claims 1, 4, 5, 7, 9, 10, 12, 13, 17-20, 22, 23, and 25

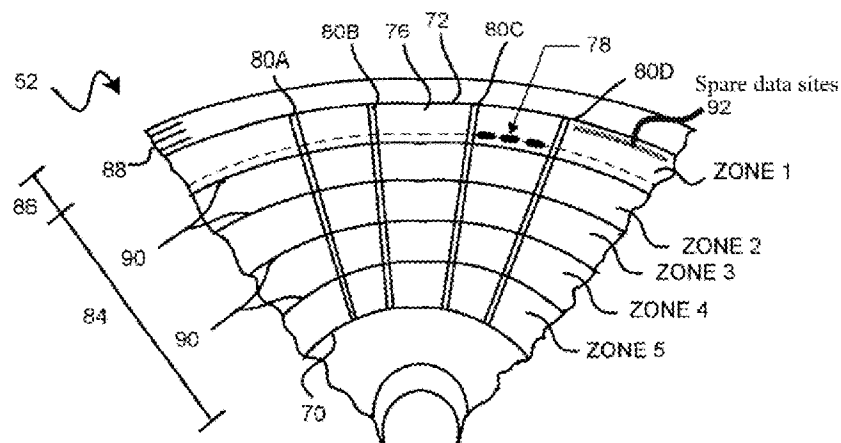
As explained below, *Schibilla* in view of *Nagar* renders obvious claims 1, 4, 5, 7, 9, 10, 12, 13, 17-20, 22, 23, and 25.

1. Overview of *Schibilla*

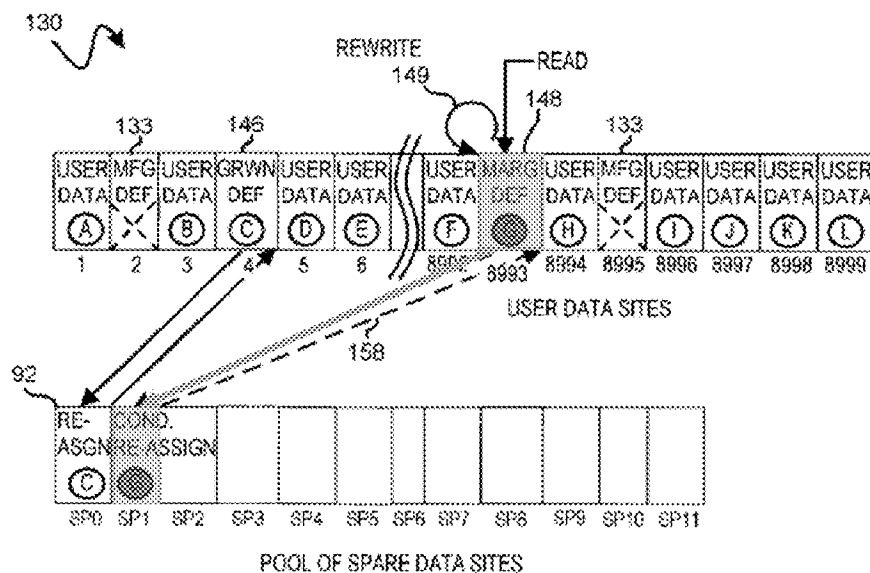
Schibilla describes techniques for monitoring a disk drive and preserving data in case of predicted failure. *See, e.g.*, Ex. PA-1, Abstract, 1:8-11, 3:20-48; Ex. C ¶ 54. In one aspect, *Schibilla* monitors read and write commands provided to data sites on a storage device. *See* Ex. PA-1, 5:23-40, 6:35-49, 7:65-8:16. During this process, *Schibilla* employs a “scoring

algorithm” to rate the “historical performance” of the device reflecting a general “‘health’ of a data site.” Ex. PA-1, 8:5-29; *see also, e.g.*, 6:36-61, 9:65-10:14; Ex. C ¶ 54.

When a data site’s performance breaches a “predefined performance threshold,” *Schibilla* performs a “conditional reassignment” process. *See* Ex. PA-1, Abstract, 4:15-41, 6:45-7:10, Figs. 3-5; Ex. C ¶ 55. There, *Schibilla* marks the data site as “marginally defective” and writes the data to a pool of spare data sites:



Ex. PA-1, 6:62-7:10, Fig. 5; Ex. C ¶ 55. This creates a “backup copy” of the data:



Ex. PA-1, 6:62-7:10, 8:30-49, Fig. 2; Ex. C ¶ 55.

Schibilla attempts to reconcile the marginally defective site by rewriting the data to the site. Ex. PA-1, 8:30-49, 10:15-41; Ex. C ¶ 56. If the rewrite process succeeds, the system resumes normal operation. Ex. PA-1, 7:48-64, 10:15-41. If not, however, the system makes the reassignment permanent. *Id.*, 8:66-9:10, 10:15-41, Figs. 3-4; Ex. C ¶ 56. *Schibilla* does so by, among other things, marking the data site as defective, updating pointer information, and logging the affected site in a list of defective sites. *See, e.g.*, Ex. PA-1, 10:15-55, Ex. C ¶ 56. This process effectively moves the affected data from the defective site to a spare data site on disk. Ex. C ¶ 56.

2. Overview of *Nagar*

Nagar discloses a Windows NT filter driver. Ex. C ¶ 57. This filter driver acts as an intermediary for intercepting commands targeted elsewhere, such as a disk driver for transferring data to disk. Ex. PA-2, 22, 33; Ex. C ¶ 57. This allows the filter driver to intercept “all I/O” between a source and “a specified mounted logical volume” (e.g., disk storage). Ex. PA-2, xvii; Ex. C ¶ 57. By intercepting requests before they reach their target, *Nagar*’s filter driver expands or replaces the functionality of the layers above or below it. Ex. PA-2, 33; Ex. C ¶ 57.

Nagar places the filter driver above a file system (e.g., NTFS or FAT) or below it:

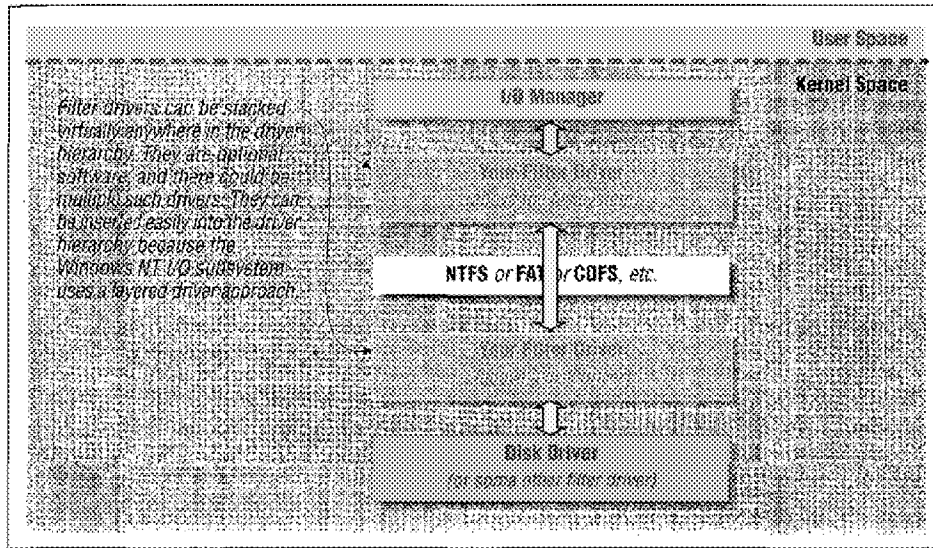


Figure 2-6. Filter drivers in the driver hierarchy

Ex. PA-2, 35; Ex. C ¶ 58. This placement enables the filter driver to intercept all information passing to and from a storage device and other computer processes:

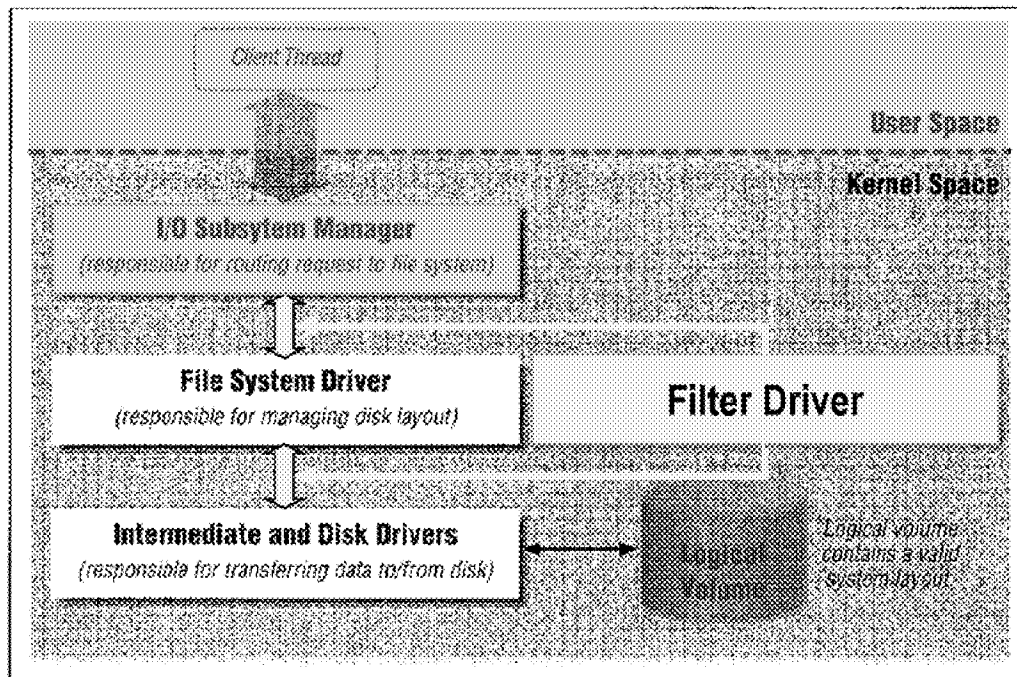


Figure 2-1. Local file system

Ex. PA-2, 22; Ex. C ¶ 58. Among other examples, *Nagar* uses filter drivers to manage disk space allocation and security, as well as error flagging and reporting. *See, e.g.*, Ex. PA-2, 86-93, 260, 619-622; Ex. C ¶ 58.

3. Public Availability of *Nagar*

Nagar published in 1997 and was publicly available before the effective filing date of the '832 patent. Ex. PA-2, 5-6. *Nagar* was catalogued in the Library of Congress as of October 23, 1997. *See id.*; *see also, e.g.*, Ex. D (Library of Congress MARC record); Ex. F ¶¶ 12-22 (librarian declaration on MARC record). *Nagar* would have been accessible and discoverable to the interested public within a few days afterward. Ex. F ¶¶ 12-22.

Indeed, those of ordinary skill located *Nagar* prior to the effective filing date of the '832 patent. For example, multiple U.S. patents predating the '832 patent cite *Nagar* and indicate its September 1997 publication date. *See, e.g.*, Exs. PA-7-PA-9. Indeed, the Board agreed that such evidence established *Nagar*'s public availability in prior IPR proceedings challenging different patents. *See, e.g., NetApp, Inc. v. KOM Software, Inc.*, IPR2019-00604, Paper 33, at 14 (P.T.A.B. Sept. 3, 2020), *aff'd*, No. 21-1181, 2021 WL 5985364 (Fed. Cir. Dec. 17, 2021); *NetApp, Inc. v. KOM Software, Inc.*, IPR2019-00608, Paper 32, at 21 (P.T.A.B. Sept. 22, 2020), *aff'd*, No. 21-1181, 2021 WL 5985364 (Fed. Cir. Dec. 17, 2021).

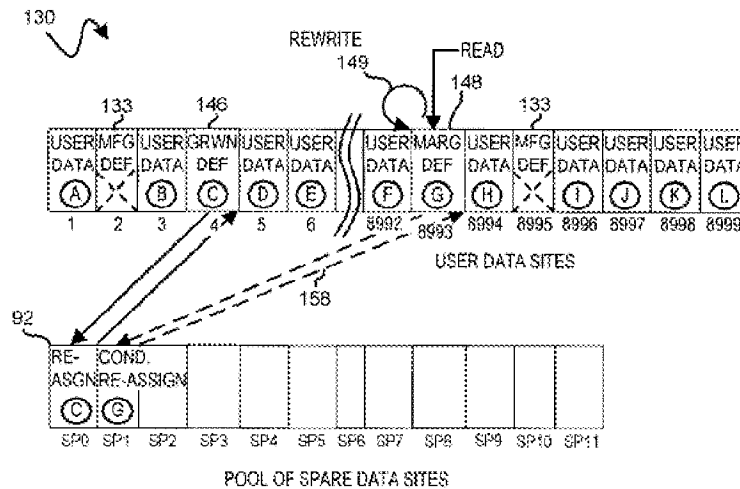
4. Independent Claim 1

a. [1pre]: “A method for monitoring performance of a storage device”

To the extent the preamble is limiting, the combination of *Schibilla* and *Nagar* (“*Schibilla-Nagar*”) teaches the preamble. Ex. C ¶ 64.

Schibilla discloses an apparatus and method for monitoring a disk drive (“*storage device*”) and using a scoring algorithm to “identify data sites that have exceeded a predefined

performance threshold.” Ex. PA-1, 6:36-49; *see also id.*, 1:8-20, 3:20-48, 8:5-29, 10:65-11:14; Ex. C ¶ 65. Data sites “exceeding the predefined performance threshold are characterized as ‘marginally defective’ sites,” and the system takes corrective action. Ex. PA-1, 6:50-7:10. This response may entail writing data from the marginally defective site to another disk location:



Id.; *see also id.* Figs. 3-5; Ex. C ¶ 65. As discussed below, a POSITA would have found it obvious to implement such disk monitoring and recovery processes as a software-based filter driver that intercepts communications between the storage device and a computer system. *See* [1a]-[1c], *infra*; Ex. C ¶ 65.

b. [1a]: “intercepting communications between a computer system and said storage device”

Schibilla-Nagar teaches this limitation. Ex. C ¶ 66. For example, *Schibilla* discloses a disk drive (“*storage device*”) containing a “disk control interface” with circuitry and processors for providing an interface between a “host system” (a “*computer system*”) and a “head disk assembly (HDA)” for executing read and write commands. Ex. PA-1, 5:23-39, Fig. 1; *see also, e.g., id.*, 5:41-56. The disk drive contains a disk control system (e.g., a controller) containing, for example, firmware for implementing the data recovery processes described below. *Id.*, 5:23-39. The host system in turn reflects “a personal computer, or other system capable of performing a

sequence of logical operations,” and may include a “microprocessor based data processing system.” *Id.*; Ex. C ¶ 66

Schibilla detects when read operations from the host computer encounter problems at a particular disk data site. *See, e.g.*, Ex. PA-1, 6:35-49, 8:5-29, 9:57-10:4; Ex. C ¶ 67. Upon detecting a problem, *Schibilla* uses a “scoring algorithm” to “identify data sites that have exceeded a predefined performance threshold.” *Id.* *Schibilla* denotes these sites “marginally defective.” *Id.*, 5:50-61, 8:17-29, 10:5-14. For marginally defective sites, *Schibilla* employs a potential data reallocation process, discussed below. *See, e.g., id.*, 6:62-7:10, 8:30-49, 10:15-41; *see also* [1b]-[1c], *infra*; Ex. C ¶ 67. While *Schibilla* monitors read and write commands from a host computer (*e.g.*, Ex. PA-1, 5:23-40) and scores the disk drive via disk drive firmware (*id.*, 9:65-10:4), *Schibilla* does not expressly disclose this processing “*intercept[s] communications between a computer system and said storage device*” with a “filter driver,” as discussed in the ’832 patent (*see* Ex. A, 3:46-53).³

Nagar discloses these details. Ex. C ¶ 68. *Nagar* is a textbook describing how to implement a “filter driver” in software that “intercepts all I/O requests targeted to a specified mounted logical volume.” Ex. PA-2, xvii. These filter drivers “intercept[] requests targeted to some existing software module,” including a “file system” or “disk driver” exchanging information with a disk (“*intercept[] communications between a computer system and said storage device*”). *Id.*, 22, 33. This enables the driver to “extend, or simply replace, the functionality provided by the original recipient of the request” via software. *Id.*, 33-34; Ex. C ¶ 68; *see* Ex. PA-2, 617-623.

³ The claims do not expressly require a “filter driver.” Nevertheless, to minimize disputes before the Office, Requester presents the grounds herein as if they do.

Nagar implements filter drivers to “intercept” a communication “before it is written out to disk” or “immediately after it has been retrieved from disk.” *Id.*, 35; *see also id.*, 118-119, 143; Ex. C ¶ 69. This enables the filter driver to “do whatever [the developer] need[s] before the request is received by a driver, or by a network driver that is asked by the file system to obtain data from secondary storage devices or across the network.” Ex. PA-2, 34. In this way, *Nagar*’s filter driver “*intercept[s] communications between a computer system and said storage device,*” as claimed:

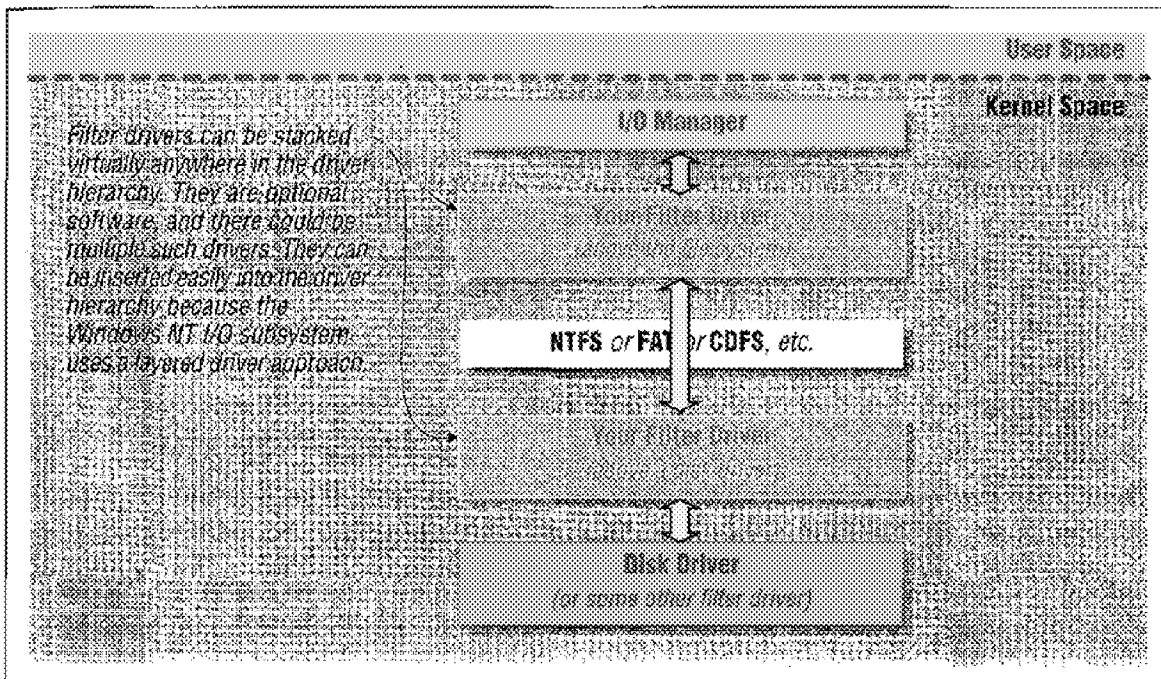


Figure 2-6. Filter drivers in the driver hierarchy

Id., 35; *see also id.*, 22, 617-622; Ex. C ¶ 69.

A POSITA would have found it obvious to implement *Schibilla*’s processing, in whole or part, within *Nagar*’s filter driver so that the filter driver would “intercept” communications (e.g., read/write commands, indications of read errors, performance metrics, etc.) between *Schibilla*’s host computer and disk drive. Ex. C ¶ 70. Specifically, a POSITA would have found it obvious

to “intercept[] all I/O” between *Schibilla*’s disk drive and host system (*e.g.*, Ex. PA-2, xvii)—including all read/write commands, data and values obtained by *Schibilla*’s disk controller, etc.—to conduct the disk monitoring and data reallocation processes described in *Schibilla* (*see* [1b]-[1c], *infra*). Ex. C ¶ 70. Even more, *Schibilla*, *Nagar*, and the ’832 patent are analogous art because they are all in the same field of endeavor—increasing the performance of storage devices over time. *See, e.g.*, Ex. PA-1, Abstract, 1:8-11, 3:20-48; Ex. PA-2, 33-34; Ex. A, 1:59-2:21; Ex. C ¶¶ 39-40, 54-58.

An ordinary artisan would have been motivated to use such a filter driver for several reasons. Ex. C ¶ 71. For example, a POSITA would have appreciated how common operating systems at the time of the ’832 patent often placed restrictions on manipulating disk memory directly. Ex. C ¶ 72. Indeed, a POSITA would have appreciated that these restrictions partially motivated the use of filter drivers in the first place. Ex. C ¶ 72. By way of example, Windows NT and 2000 (released in 1993 and 2000, respectively) often limited ways of allocating or manipulating memory on disk to certain allowed processes (*e.g.*, “privileged” mode). Ex. C ¶ 72. Given this knowledge, a POSITA would have appreciated how *Nagar*’s filter driver would enable developers to perform these disk processes—such as those discussed in *Schibilla*—at the driver level without such restrictions. Ex. C ¶ 72. *Nagar* itself discloses an example where a filter driver “[i]ntercept[s]” all read and write requests for data “stored to disk” for security purposes. Ex. PA-2, 618. Accordingly, a POSITA would have been motivated to implement *Schibilla*’s processes within a filter driver to extend a disk controller’s processing, make disk allocation simpler, and avoid having to potentially rely on permitted access modes to manipulate storage. Ex. C ¶ 72.

A POSITA would also have appreciated how *Nagar*'s filter driver would streamline disk management processing and development. Ex. C ¶ 73. For example, a POSITA would have appreciated updating *Nagar*'s filter driver to implement *Schibilla*'s disk management routines would be simpler than updating the firmware for various types of disk drives separately to accomplish the same task. Ex. C ¶ 73.

Indeed, *Nagar* expressly teaches these motivations. Ex. C ¶ 74. *Nagar* explains, for instance, how filter drivers “extend, or simply replace,” the underlying functionality of targets like *Schibilla*'s disk controller. Ex. PA-2, 33-34; Ex. C ¶ 74. As *Nagar* teaches, filter drivers allow developers to “extend” the functionality of their storage devices while obviating the need to “write a completely new file system driver” or update a disk's firmware. Ex. PA-2, 33-34; Ex. C ¶ 74. A POSITA would have understood this filter driver would be implemented “without having to redesign all other existing Windows NT file system, disk, and other intermediate drivers.” Ex. PA-2, 35; Ex. C ¶ 74. Accordingly, a POSITA would have been motivated to implement *Schibilla*'s processing via *Nagar*'s filter driver to extend the capabilities of a storage disk (and disk driver) without having to redesign a file system, disk driver, or other firmware. Ex. C ¶ 74. Indeed, *Nagar* provides an express motivation of using a filter driver to perform hierarchical storage management to efficiently allocate data among storage media. Ex. PA-2, 620-622, Ex. C ¶ 74. And other references at the time of the '832 patent already disclosed implementing disk failure predicting algorithms at the filter driver level. *See, e.g.*, Ex. C ¶ 74; PA-10, Abstract, 3:31-40.

A POSITA would have had a reasonable expectation of success in implementing *Schibilla*'s processes as a filter driver, as in *Nagar*. Ex. C ¶ 75. The combination merely employs *Schibilla*'s well-known disk analysis on monitoring methods (*see* [1b]-[1c], *infra*), using well-

understood filter driver technology (as disclosed in *Nagar*) to achieve predictable results. Ex. C ¶ 75. As discussed above, for example, ordinary artisans had already implemented disk failure prediction processes as filter drivers at the time of the '832 patent. *See* Ex. C ¶ 75. Accordingly, using a filter driver to intercept communications (as in *Nagar*) and to implement *Schibilla*'s known memory allocation processes would have reflected nothing more than combining existing elements in known ways to achieve predictable results. Ex. C ¶ 75.

c. [1b]: “analyzing said intercepted communications relative to a threshold value for the performance of said storage device”

Schibilla-Nagar teaches this element. Ex. C ¶ 76. As discussed above, *Schibilla-Nagar* teaches “*intercept[ing]*” “all I/O” between a storage device and a host system. *See* [1a], *supra*. In the proposed combination, *Schibilla* discloses “*analyzing*” such intercepted communications “*relative to a threshold value for the performance of said storage device.*” Ex. C ¶ 76.

Specifically, *Schibilla* uses a scoring algorithm and performance thresholds to determine when user data sites contain marginal defects. *E.g.*, Ex. PA-1, 6:35-61; Ex. C ¶ 77. For example, when a “disk read operation” requires “multiple retries” to successfully read data from a data site, *Schibilla* determines whether the data site is “marginally defective.” Ex. PA-1, 6:35-61; Ex. C ¶ 77. To do so, *Schibilla* employs a “scoring algorithm” to “identify data sites that have exceeded a predefined performance threshold (e.g., the number of read retries)” (i.e., “*analyzing [intercepted I/O] to a threshold value for the performance of said storage device*”). Ex. PA-1, 6:35-61; Ex. C ¶ 77. When a data site “exceed[s] the predefined performance threshold,” *Schibilla* teaches the system deems it “marginally defective” and conducts further processing. Ex. PA-1, 6:50-7:10.

Schibilla's scoring algorithm “provides a general indication of the ‘health’ of a data site.” Ex. PA-1, 8:17-29. The algorithm “analyzes [the] historical performance information for the data

site,” such as “the number of ‘read errors’” or “the number of retries required to read the data block,” to make this determination. *Id.*; Ex. C ¶ 78. The “performance metric from the data site is then compared against a predefined performance threshold in order to determine if the data site is marginally defective.” Ex. PA-1, 8:17-29; *see also, e.g., id.*, 6:36-61, 9:57-10:14 (explaining how “[t]he scoring algorithm looks at performance attributes such as the number of retries required to read information from a data site” to identify potentially compromised data sites), claims 1, 3-4, 12; Ex. C ¶ 78. The ’832 patent describes its performance metrics and thresholds in the same way. *See, e.g.,* Ex. A, 5:55-6:2, 6:16-36, Fig. 3.

As described above for [1a], a POSITA would have found it obvious to implement *Schibilla*’s processing, in whole or part, within *Nagar*’s filter driver. *See* [1a], *supra*. This includes implementing *Schibilla*’s scoring algorithm and performance thresholds within *Nagar*’s filter driver. Ex. C ¶ 79. As described above, this implementation simply applies *Schibilla*’s disk analysis to well-known filter driver technology. *See* [1a], *supra*; Ex. C ¶ 79.

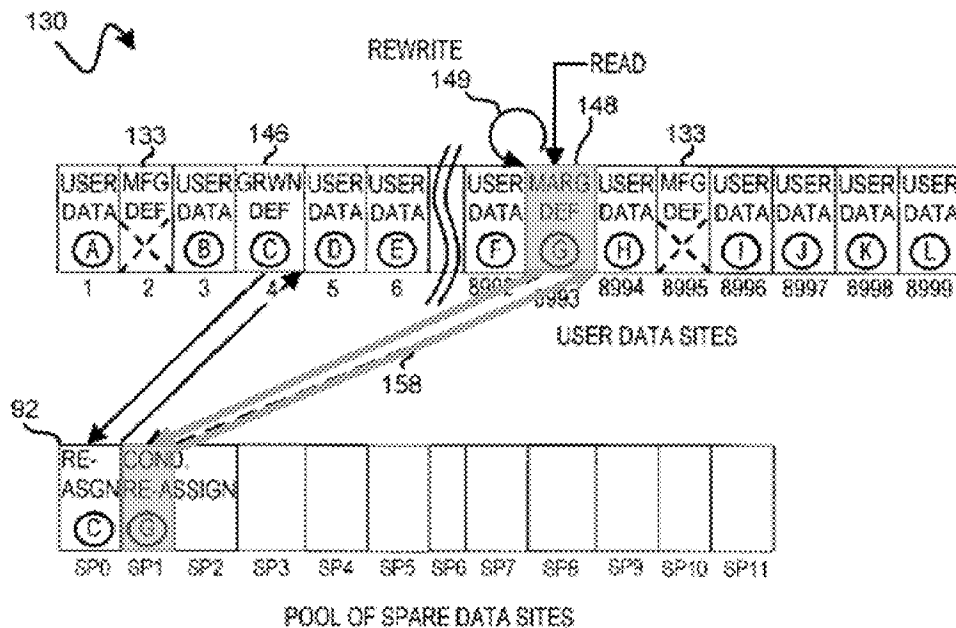
Because *Schibilla* analyzes the performance of a data site against a predefined threshold, a POSITA would have understood *Schibilla-Nagar* to teach “*analyzing said intercepted communications relative to a threshold value for the performance of said storage device.*” Ex. C ¶ 80.

- d. **[1c]: “responding to a decline in the performance of said storage device based on said analyzed intercepted communications by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device”**

Schibilla-Nagar teaches this element. Ex. C ¶ 81. As discussed above, *Schibilla* in view of *Nagar* teaches “*analy[zing] intercepted communications*” (*see* [1a], *supra*) and “*responding to a decline in the performance of said storage device*” (*see* [1b], *supra*). Ex. C ¶ 81. For example, *Schibilla* detects “*a decline in the performance of said storage device*” when its scoring

algorithm determines the “performance” of a data site crosses a “predefined storage threshold.”
Ex. PA-1, 6:36-61, 8:17-29, 9:65-10:14, claims 1, 3-4, 12; Ex. C ¶ 81; [1b], *supra*.

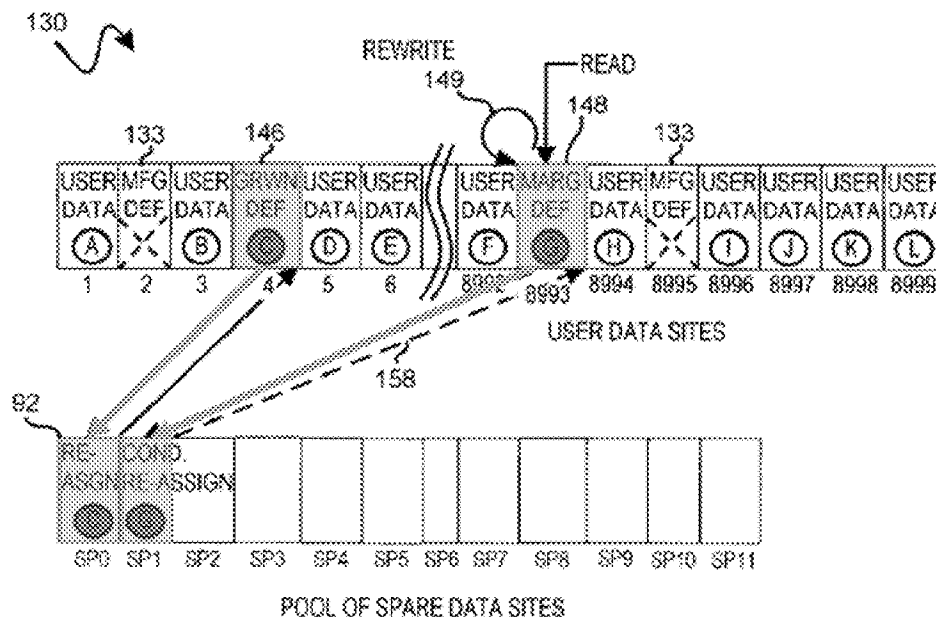
Schibilla also “respond[s]” to this drop in performance “by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device.”
Ex. C ¶ 82. For example, when *Schibilla* encounters a marginally defective data site (defined by its scoring algorithm), *Schibilla* “recovers data from the marginally defective data site [by] writ[ing] the recovered data to a spare data site elsewhere on the disk surface.” Ex. PA-1, 6:62-7:10. This rewrite process involves “writing the data from the marginally defective data site to an alternative location” to “provide[] a backup copy of the data”:



Id., 6:62-7:10, Fig. 5; Ex. C ¶ 82.

Schibilla then attempts to rewrite the data back onto the marginally defective site in a process called “conditional reassignment.” Ex. PA-1, Abstract, 4:15-34, 6:62-7:47, 8:50-65, 10:15-41, Figs. 4-5; Ex. C ¶ 83. If the rewrite procedure succeeds, the system continues normal

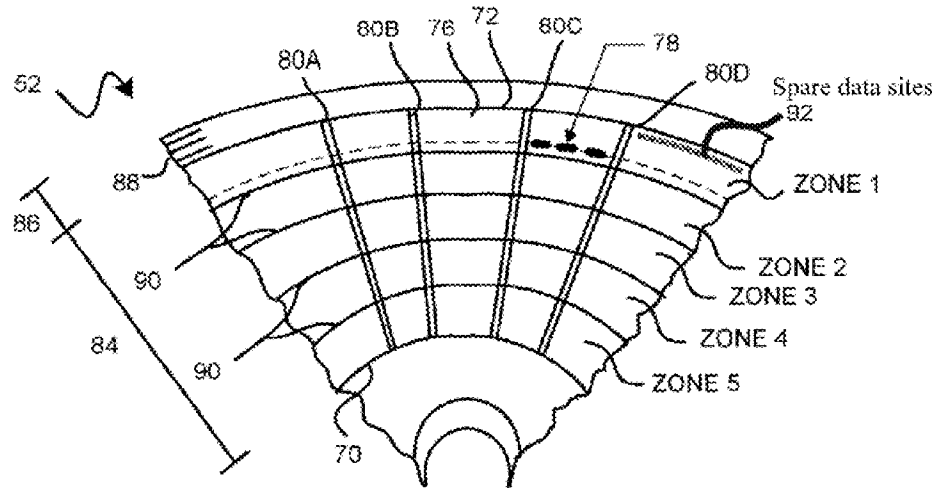
operation. *Id.*, 7:48-64, 10:15-41. If not, the system makes the reassignment permanent. *Id.*, 8:66-9:10, 10:15-41, Figs. 3-4; Ex. C ¶ 83. This process involves, among other things, updating pointer information, logging the defective data site in a cross-reference table, and incrementing pointers to the next available location in the pool of spare data sites. *See* Ex. PA-1, 8:66-9:10, 10:15-41, 10:56-11:2, Fig. 5; Ex. C ¶ 83. *Schibilla's* conditional reassignment thus moves data from defective sites to a pool of spare data sites, such as moving data blocks C and G from original data sites 4 and 8993 to spare data sites SP0 and SP1:



Ex. PA-1, Fig. 5; Ex. C ¶ 83.

A POSITA would have understood *Schibilla's* conditional reassignment process “*automatically reallocat[es] at least some data on said storage device.*” Ex. C ¶ 84. The ’832 patent describes reallocation as merely recording data to a different sector on disk. *See, e.g.*, Ex. A, 2:41-47, 2:58-64, 8:30-37, 8:59-65, 9:47-62, 10:16-21. Thus, a POSITA would have

understood how *Schibilla*'s conditional reassignment, which “reassign[s data] to a spare data site at an alternative location, as shown at 92,” reflects the claimed “*reallocat[ion]*”:



Ex. PA-1, 10:5-13, Figs. 1, 5; Ex. C ¶ 84; *see also* Ex. PA-1, 6:62-7:10.

A POSITA would also have understood *Schibilla*'s conditional reassignment "*enhance[s]* *continued operation of the storage device*." Ex. C ¶ 85. *Schibilla*'s disk continues to operate after the assignment. *See, e.g.*, Ex. C ¶ 85; Ex. PA-1, 3:21-48, 4:55-62, 6:50-7:10 (explaining how conditional reassignment can "provid[e] a performance enhancement"), 8:50-9:10. For example, *Schibilla* describes hosting a table of defective data sites—identified via the above process—marking sites to avoid or further examine during subsequent read and write operations. *See, e.g.*, Ex. PA-1, 9:27-55, 10:15-55, Fig. 5; Ex. C ¶ 85. This "*enhance[s]*" the disk drive by enabling it to "*continue[] operat[ing]*" without failure and by "minimiz[ing any] operational inefficiencies." Ex. PA-1, Abstract, 1:57-2:14, 3:14-17; Ex. C ¶ 85. These disclosures comport with how the Applicant discussed this limitation during prosecution. *See, e.g.*, Ex. C ¶ 85; Ex. B at 283 (distinguishing this limitation from decommissioning an entire disk from a network).

Accordingly, because *Schibilla* reassigns data in response to detecting a decline in disk performance, a POSITA would have understood *Schibilla-Nagar* to teach “*responding to a decline in the performance of said storage device based on said analyzed intercepted communications by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device.*” Ex. C ¶ 86.

5. Claim 4

Claim [4] depends from claim 1 and recites “*wherein intercepting said communications comprises intercepting an error reported by said storage device.*” Ex. A, claim 4. *Schibilla-Nagar* teaches this additional limitation in claim 4. Ex. C ¶ 87.

As discussed above, *Schibilla* detects when there has been a “read error” for a particular data site on a storage device. *See* Ex. PA-1, Abstract, 4:15-34, 6:50-61, 8:5-29, Figs. 3-4; Ex. C ¶ 88; *see also* Ex. PA-1, claims 1, 13; [1b], *supra*. For example, when “an error is encountered in [a] read operation,” the system may employ a “read error recovery procedure” to recover the data from the affected block as discussed above. *See* Ex. PA-1, 8:5-29; *see also* [1b]-[1c], *supra*. In some embodiments, the system uses a scoring algorithm, such as “the number of ‘read errors,’” to determine when to implement this procedure. Ex. PA-1, 8:5-29; Ex. C ¶ 88. A POSITA would have thus understood *Schibilla* to disclose “*an error reported by said storage device.*” Ex. C ¶ 88. In combination with *Nagar*, *Schibilla-Nagar* would “*intercept[]*” these errors reported by the storage device to conduct the foregoing read-error recovery procedure:

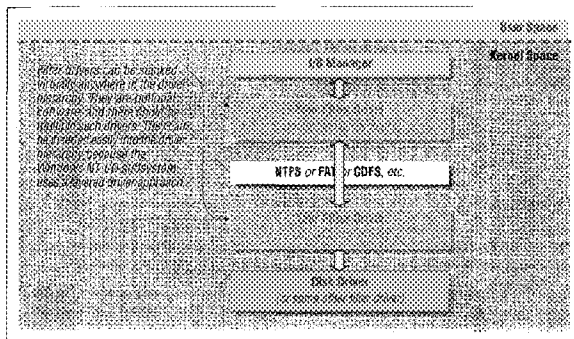


Figure 2-6. Filter drivers in the driver hierarchy

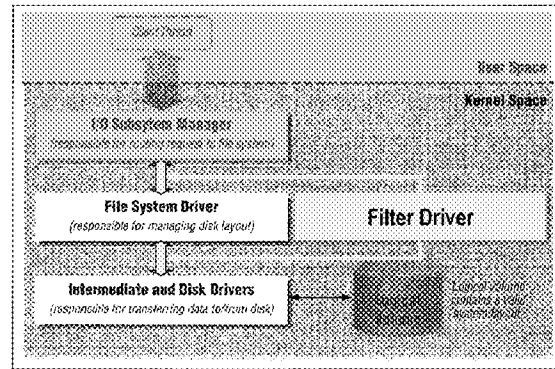


Figure 2-7. Local file systems

Ex. PA-2, 22, 35; Ex. C ¶ 88; *see also* [1a], *supra*.

It would have been obvious to a POSITA to intercept errors reported by *Schibilla*'s disk drive at *Nagar*'s filter driver to conduct the data recovery process described for claim 1. *See* [1a]-[1c], *supra*. *Nagar* teaches how it is the driver's duty to handle errors when a disk drive is not functioning properly (as in *Schibilla*'s read errors). *See* Ex. PA-2, 65-66; Ex. C ¶ 89. Indeed, the main purpose of *Nagar*'s filter driver is to “extend” or “replace” the functionality of the disk drive. Ex. PA-2, xvii, 33-34. Because *Schibilla*'s disk drive detects performance problems, such as read errors, for particular data sites, a POSITA would have found it obvious to “*intercept[]*” these “*error[s] reported by said storage device*” with a filter driver, as in *Nagar*. Ex. C ¶ 89.

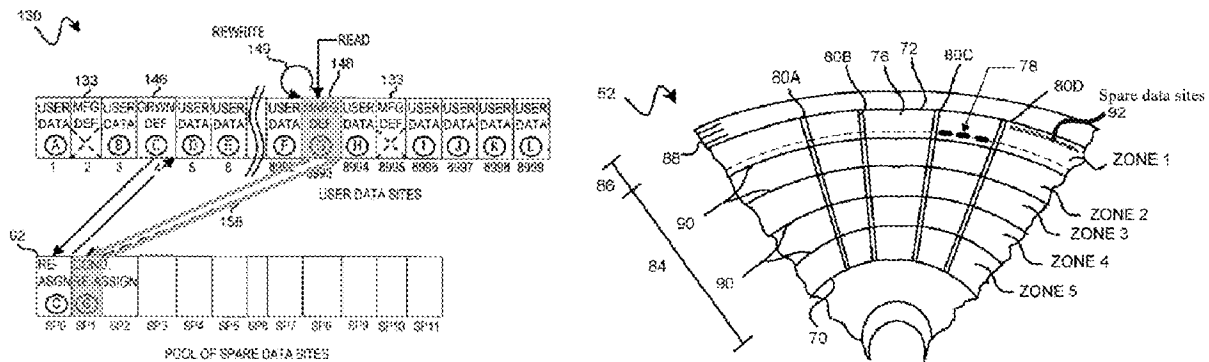
A POSITA would have had a reasonable expectation of success in intercepting such errors, as doing so would simply combine known functions to achieve predictable results. Ex. C ¶ 90. As discussed above, *Schibilla* already discloses determining errors in a disk drive. *See supra*; *see also* [1b]-[1c], *supra*. Likewise, *Nagar* teaches “intercepting” communications—including “all I/O” to and from a disk drive. *See* [1a], *supra*. A POSITA would have thus had a reasonable expectation of success in intercepting errors detected and reported by a disk drive via a filter driver, as doing so would simply combine existing methods (e.g., read error detection, filter drivers) in known ways (filter driver monitoring disk drives) to achieve predictable results.

Ex. C ¶ 90. Indeed, prior art at the time of the '832 patent already combined these functions, disclosing implementing drive failure algorithms via filter drivers. *See, e.g.*, Ex. C ¶ 90; Ex. PA-10, 3:31-51, 6:66-7:46.

6. Claim 5

Claim [5] depends from claim 1 and recites “*determining an access location on said storage device and an access frequency for data stored thereon, based on said intercepted communications.*” Ex. A, claim 5. *Schibilla-Nagar* teaches this additional limitation of claim 5. Ex. C ¶ 91.

For example, *Schibilla* discloses determining an “access location” for read or write operations by identifying particular “site[s]” those operations reference. *See* [1pre]-[1c], *supra*; Ex. C ¶ 92; *see also, e.g.*, Ex. PA-1, Abstract, 6:36-7:10, 8:5-16, Figs 3-5. This enables *Schibilla* to monitor the performance of a particular “site” or “address” in data. *See, e.g.*, Ex. PA-1, 9:65-10:55. *Schibilla* thus monitors operations performed on a site-by-site basis:



Ex. PA-1, Figs. 1, 5; Ex. C ¶ 92. A POSITA would have understood that *Schibilla*’s monitoring of particular sites on disk reflects the claimed “access location on said storage device.” Ex. C ¶ 92.

Nagar likewise teaches determining an “*access frequency for data stored thereon*.” Ex. C ¶ 93. For example, *Nagar* teaches how its filter driver may provide hierarchical storage management (HSM) tasks. *See* Ex. PA-2, 260-261, 620-622; Ex. C ¶ 93. HSM reflects the practice of “automatically transferring infrequently accessed data to slower, cheaper media,” while frequently accessed data resides in faster locations. *See* Ex. PA-2, 260-261, 620-222 (explaining how HSM “involves automatic transfer of infrequently used data to slower but cheaper secondary storage media and an automatic transfer back to regular storage if the migrated data files are accessed”); Ex. C ¶ 93. Because *Nagar* allocates data based on access frequency, a POSITA would have understood *Nagar* to teach determining an “*access frequency*” of data, as claimed. Ex. C ¶ 93.

As discussed for claim 10 below (*see* [10], *infra*), a POSITA would have found it obvious to combine determining an access location (as in *Schibilla*) with an access frequency (as in *Nagar*) to improve data storage efficiency. Ex. C ¶ 94. For example, *Nagar*’s HSM discussion teaches offloading infrequently accessed data for efficiency purposes. *See, e.g.*, Ex. PA-2, 260, 620-622. A POSITA would thus have found it obvious to determine an access location and frequency of data to identify sites of high-frequency access. Ex. C ¶ 94. Indeed, *Schibilla* monitors the “historical performance” of a hard drive data site (Ex. PA-1, 8:17-29), and its reallocation process attempts to rewrite data in a way preserving contiguity with related data for better efficiency (*see id.*, 2:15-3:4, 7:23-47, 8:50-65, 10:15-55). Given these disclosures, a POSITA would have found it obvious to monitor not just the “site” but the “frequency” of data access (e.g., for HSM purposes) to improve the efficiency and reallocation of data in *Schibilla-Nagar*. Ex. C ¶ 94.

A POSITA would have had a reasonable expectation of success in combining the determination of a data access “*location*” (as in *Schibilla*) with a data access “*frequency*” (as in *Nagar*). Ex. C ¶ 95. As explained, existing HSM methods already examined the frequency of data usage. *See* Ex. PA-2, 260-261, 620-222. Combining these teachings with *Schibilla* would have thus amounted to nothing more than the application of known methods (e.g., HSM methods) to existing technology (*Schibilla*’s data site monitoring and reallocation) to achieve predictable results (e.g., monitoring both data access locations *and* frequency). Ex. C ¶ 95.

7. Claim 7

Claim [7] depends from claim 1 and recites “*logging said communications over time.*” Ex. A, claim 7. *Schibilla-Nagar* teaches this additional limitation of claim 7. Ex. C ¶ 96.

Nagar teaches how operating systems, such as Windows NT, use an “Event Log” in conjunction with a filter driver that “serves as a central repository for messages sent by various software modules on that machine.” Ex. PA-2, 87; Ex. C ¶ 97; *see generally* Ex. PA-2, 86-93 (describing event logging). This log includes information such as “error messages possibly caused by software or malfunctioning of attached hardware peripherals, warning messages that might indicate recovered errors or the possibility of an impending error, and informational (or status) messages indicating that some important activity had transpired.” Ex. PA-2, 86-87; Ex. C ¶ 97. *Nagar* explains how a “kernel-mode driver,” such as its filter driver, “logs an event indicating that something significant has occurred.” Ex. PA-2, 86-87; *see also* Ex. PA-2, 389 (discussing “logging file system implementations” to “log file streams”), 617-618 (explaining how a “filter driver is a kernel-mode driver”). For example, the event log “allows the kernel module to include other pertinent information for events indicating an error condition, such as the number of times [an] operation has been retried” and other I/O status information. *See* Ex. PA-2, 86-87; Ex. C ¶ 97.

In addition to the reasons to intercept communications between a disk and a host system in view of *Schibilla* and *Nagar* (see [1a], *supra*), a POSITA would have been motivated to log intercepted communications in an event log as *Nagar* teaches. Ex. C ¶ 98; Ex. PA-2, 86-93. As discussed, *Nagar* logs events, such as error messages from a disk and the “number of times [an] operation has been retried.” Ex. PA-2, 86-87. A POSITA would have appreciated the benefits of implementing this logging scheme with *Schibilla*, which uses “the number of read retries” as a scoring metric to identify potentially compromised data sites on a disk drive. Ex. PA-1, 6:36-61; *see also* Ex. PA-1, 8:17-29. *Schibilla* further describes logging data sites with potential defects and memorializing data reassignments in a cross-reference table. *See* Ex. PA-1, 9:11-64, 10:36-55, Fig. 5, claims 1, 14; Ex. C ¶ 98.

Given these disclosures, a POSITA would have been motivated to log all I/O commands, error messages, defective site lists, reassignment cross-references, and other events described in *Schibilla*—intercepted between a computer system and a disk drive (as in *Nagar*)—to construct a responsive and robust filter driver as *Nagar* instructs. Ex. C ¶ 99; *see also* Ex. PA-2, 86-93; [1a], *supra*. A POSITA would have appreciated how doing so would create an event log for, among other things, determining the “health” of the monitored disk drive, computing its performance metrics, and relaying such information “to the system administrator or to the user of the host machine.” *See* Ex. PA-1, 8:17-29; Ex. PA-2, 86-90; Ex. C ¶ 99. For example, a POSITA would have been motivated to log all I/O, performance calculations, error messages, conditional reassignment information (as in *Schibilla*) to potentially provide helpful “errors messages possibly caused by software or malfunctioning” drives (as in *Nagar*) to users and administrators. Ex. PA-2, 86-90; Ex. C ¶ 99. And a POSITA would have been motivated to log “all I/O” requests targeted to *Schibilla*’s disk drive (see Ex. PA-2, xvii) to, for example, create a log from

which to track and derive performance scores, generate error messages, and monitor system performance (*see, e.g.*, Ex. PA-1, 8:17-29, 9:65-10:14). Ex. C ¶ 99.

A POSITA would have had a reasonable expectation of success in logging such intercepted events. Ex. C ¶ 100. Indeed, *Nagar* provides examples of how to do so with illustrative data structures and event identifiers. *See* Ex. PA-2, 86-93. Thus, combining a communications log with *Schibilla* would have involved nothing more than applying known methods (e.g., *Nagar*'s communications log) to existing technology (e.g., the existence of I/O commands and reported errors in *Schibilla*) to yield predictable results (logging intercepted communications). Ex. C ¶ 100.

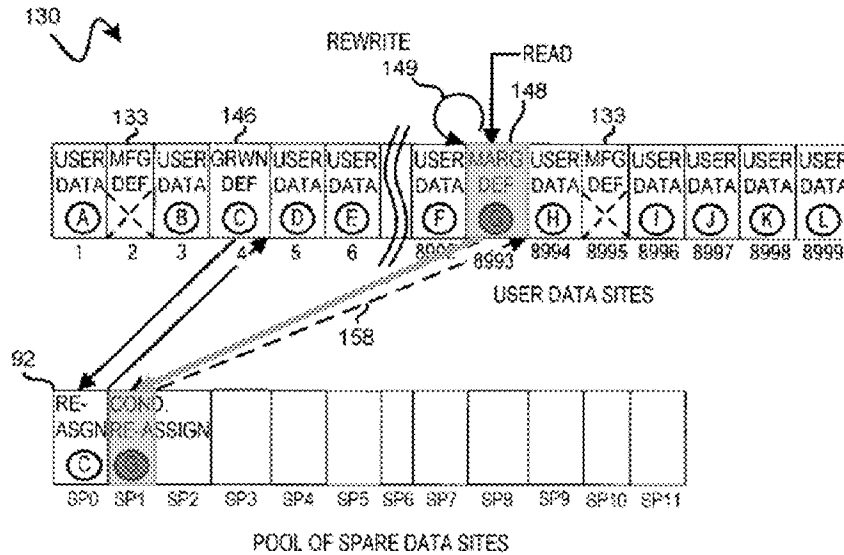
8. Claim 9

Claim [9] depends from claim 1 and recites “*wherein responding to said declining performance of said storage device further comprises automatically backing-up data stored on said storage device.*” Ex. A, claim 9. *Schibilla-Nagar* teaches this additional limitation of claim 9. Ex. C ¶ 101.

Schibilla explains how its automatic conditional reassignment creates a “backup copy of the data” on the disk (“*automatically backing-up data stored on said storage device*”). Ex. PA-1, 6:62-7:10, 8:30-49; *see also* [1c], *supra*; Ex. C ¶ 102. As discussed above, this conditional reassignment occurs when a disk’s performance breaches a threshold (“*responding to said declining performance of said storage device*”). *See* [1b]-[1c], *supra*; Ex. C ¶ 102.

By copying the impacted data to a separate portion on the drive, *Schibilla*’s conditional reassignment “cop[ies] the data block to [a] spare data site,” which “provid[es] a data backup should a failure occur during the rewrite operation to the marginally defective data site.” Ex. PA-1, 8:30-49; *see also, e.g., id.*, 6:62-7:10, 10:15-41; Ex. C ¶ 103; [1c], *supra*. Before the conditional reassignment completes, data in marginally defective sites exist in both the original

site and the pool of spare data sites, such as data block G being recorded in data sites 8993 and SP1 below:



Ex. PA-1, Fig. 5; Ex. C ¶ 103. Because this conditional reassignment “cop[ies]” data to a spare site to “provid[e] a data backup” of the data block, a POSITA would have understood *Schibilla* to disclose “*automatically backing-up data stored on said storage device.*” Ex. C ¶ 103.

To the extent Patent Owner disputes *Schibilla*’s disclosures in this regard, it would have been obvious to a POSITA to automatically “*backup*” data in the face of declining performance. Ex. C ¶ 104. For example, a POSITA would have found it obvious to automatically backup at least a portion of data in *Schibilla*’s disk drive (e.g., data stored in marginally defective sites and other contiguous data) in response to detecting that the drive’s performance crossed some performance threshold, as *Schibilla* discloses. Ex. C ¶ 104. At the time of the ’832 patent, it was common and well-known to “*automatically back[]-up data*” in response to declining performance in disk drives. *See* Ex. C ¶ 104. A POSITA would therefore have found it obvious, in light of *Schibilla-Nagar* and this background knowledge, to back up data in response to

detecting a performance decline (e.g., a potential device failure) to storage to preserve data and avoid data loss. Ex. C ¶ 104.

A POSITA would have been motivated to automatically backup data in response to performance drops for several reasons. Ex. C ¶ 105. For example, a POSITA would have appreciated how data backups, such as copying data to spare drives, would help prevent data loss due to power interruptions or failing drives. Ex. C ¶ 105. Indeed, *Schibilla* itself teaches this motivation. Ex. C ¶ 105. A POSITA would have understood that poor disk drive performance might signal a failing drive and thus would have been motivated (and found it obvious) to copy data from a failing drive into a more secure location. Ex. C ¶ 105. Indeed, many references at the time of the '832 already disclosed backing up data in response to a predicted drive failure. *See* Ex. C ¶ 105; Ex. PA-12, Abstract; Ex. PA-13, Abstract. Thus, a POSITA would have been motivated to apply these techniques to *Schibilla-Nagar* to help achieve *Schibilla*'s stated goal of preventing data loss in the case of an expected drive malfunction. Ex. C ¶ 105.

It would have been within the skill level of a POSITA to backup data in response to performance issues. Ex. C ¶ 106. As discussed, *Schibilla* already teaches monitoring disk drives, detecting waning performance, and reallocating and backing up data in response. *See, e.g.*, [1a]-[1c], *supra*. And as explained, backing up data was a prevalent feature of the art at the time of the '832 patent. Ex. C ¶ 106. Backing up data in alternative locations in response to declining disk performance would have thus reflected nothing more than the application of known solutions (e.g., backing up data, monitoring disk performance) to existing methods (e.g., disk performance monitoring and data management methods) to yield predictable results. Ex. C ¶ 106.

9. Claim 10

Claim [10] depends from claim 1 and recites “*wherein reallocating at least some data on said storage device is based on usage patterns of said data.*” Ex. A, claim 10. *Schibilla-Nagar* teaches this additional limitation of claim 10. Ex. C ¶ 107.

As discussed above, *Nagar* teaches how its filter driver may provide hierarchical storage management (HSM) tasks. *See* Ex. PA-2, 260-261, 620-622; Ex. C ¶ 108; *see also* [5], *supra*. As *Nagar* teaches, HSM reflects the practice of “automatically transferring infrequently accessed data to slower, cheaper media,” while frequently accessed data resides in faster locations. Ex. PA-2, 260-261, 620-222 (explaining how HSM “involves automatic transfer of infrequently used data to slower but cheaper secondary storage media and an automatic transfer back to regular storage if the migrated data files are accessed”). To do so, the HSM filter driver intercepts requests to access infrequently used data, transfers data back to a local file system, and then forwards the request to the file system driver. Ex. PA-2, 621-622; Ex. C ¶ 108. Because *Nagar* teaches allocating data to specific disks based on a frequency of access, a POSITA would have understood *Nagar* to teach “*reallocating at least some data on said storage device . . . based on usage patterns of said data.*” Ex. C ¶ 108.

A POSITA would have been motivated to reallocate data based on historical usage to provide more efficient access to the reallocated data. Ex. C ¶ 109. Indeed, both *Schibilla* and *Nagar* teach this concept. *Id.* For example, *Schibilla* discloses monitoring the “historical performance” of a hard drive data site (Ex. PA-1, 8:17-29), and its reallocation process attempts to rewrite data in a way to preserve contiguity with related data for better efficiency (*see id.*, 2:15-3:4, 7:23-47, 8:50-65, 10:15-55). *Nagar*’s HSM discussion likewise teaches offloading infrequently accessed data for efficiency purposes. *See* Ex. PA-2, 260, 620-622. Given these teachings, a POSITA would have found it obvious in view of *Schibilla-Nagar* to reallocate data,

such as reallocating data to *Schibilla*'s spare data sites, to keep data frequently accessed together to keep them as contiguous as possible to improve drive performance. Ex. C ¶ 109.

A POSITA would have had a reasonable expectation of success in relocating data based on historical usage. Ex. C ¶ 110. As explained, *Nagar* already teaches monitoring the frequency with which a user accesses data. *See* Ex. PA-2, 260, 620-622. And *Schibilla* teaches the concepts of data contiguity and monitoring historical performance of data sites. *See* Ex. PA-1, 2:15-3:4, 7:23-47, 8:50-65, 10:15-55. Accordingly, a POSITA would have appreciated how *Schibilla-Nagar* would merely apply known concepts (e.g., HSM, data contiguity) to existing processes (e.g., filter drivers and *Schibilla*'s reallocation method) to achieve predictable results (e.g., reallocating data based on historical use). Ex. C ¶ 110.

10. Independent Claim 12

Schibilla-Nagar renders obvious independent claim 12. Ex. C ¶ 111.

a. [12pre]: “An apparatus for monitoring performance of a storage device”

Schibilla-Nagar teaches this preamble for the reasons discussed in [1pre]. *See* [1pre], *supra*; Ex. C ¶ 112.

b. [12a]: “computer readable storage media”

Schibilla-Nagar teaches this limitation for the reasons discussed in [1pre]-[1c]. *See* [1pre]-[1c], *supra*; Ex. C ¶ 113. For example, a POSITA would have appreciated how *Schibilla*'s disk drive and host system reflect or otherwise include a computer-readable storage medium storing, for example, software and firmware executable by a processor. *See* Ex. C ¶ 113; Ex. PA-1, 5:23-40, 9:27-35, 9:65-10:4, 10:42-55, Fig. 1. And a POSITA would have appreciated how *Nagar*'s filter drivers reflect computer-readable software code for intercepting operations between a host and a logical volume. Ex. C ¶ 113; Ex. PA-2, xvii, 22-24, 34-36.

c. **[12b]: “computer readable program code stored on said computer readable storage media”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [12a]. *See* [12a], *supra*; Ex. C ¶ 114.

d. **[12b(a)]: “program code for intercepting communications between a computer system and said storage device”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1a]. *See* [1a], *supra*; Ex. C ¶ 115.

e. **[12b(b)]: “program code for analyzing said communications, wherein said communications are compared to a predicted decline in performance of said storage device”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1a]-[1c]. *See* [1a]-[1c], *supra*; Ex. C ¶ 116. For example, a POSITA would have appreciated how *Schibilla*’s disclosure of comparing disk performance relative to a “predefined performance threshold” (*e.g.*, Ex. PA-1, 6:36-61) and taking corrective actions when the disk drive crosses that threshold reflects “*a predicted decline in performance of said storage device.*” Ex. C ¶ 116. In the proposed combination with *Nagar*’s filter driver, *Schibilla-Nagar* would undertake these actions upon “*analyzing*” intercepted communications as discussed for claim 1. *See* [1a]-[1c], *supra*; Ex. C ¶ 116.

f. **[12b(c)]: “program code for responding to a decline in the performance of said storage device by backing up at least some data on said storage device prior to said predicted failure of said storage device to enhance continued operation of said storage device”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1a]-[1c] and [9]. *See* [1a]-[1c], [9], *supra*; Ex. C ¶ 117. For example, *Schibilla-Nagar* teaches “*responding to a decline in the performance of said storage device ... prior to said predicted failure of said storage device to enhance continued operation of said storage device*” for the reasons discussed

in [1b]-[1c]. *See* [1b]-[1c], *supra*; Ex. C ¶ 117. And as explained in [9], this response may include “*backing up at least some data on said storage device.*” *See* [9], *supra*; Ex. C ¶ 117.

11. Claim 13

Claim [13] depends from claim 12 and recites “*program code for intercepting an error reported by said storage device.*” Ex. A, claim 13. *Schibilla-Nagar* teaches this limitation for the reasons discussed in [4]. *See* [4], *supra*; Ex. C ¶ 118.

12. Claim 17

Claim [17] depends from claim 12 and recites “*a communications log,*” “*program code for logging said communications over time in said communications log,*” and “*program code for determining the decline in performance of said storage device based at least in part on said logged communications.*” Ex. A, claim 17. *Schibilla-Nagar* teaches this limitation for the reasons discussed in [1a]-[1c] and [7]. *See* [1a]-[1c], [7], *supra*; Ex. C ¶ 119.

For example, as discussed in [7], *Schibilla-Nagar* teaches “*logging said communications over time in [a] communications log.*” *See* [7], *supra*. And as discussed in [1a]-[1c] and [7], *Schibilla-Nagar* would have “*determin[ed] the decline in performance of said storage device based at least in part on said logged communications.*” Ex. C ¶ 120. In the proposed *Schibilla-Nagar* combination, for example, a POSITA would have found it obvious to determine *Schibilla*’s disk drive crossed a predefined performance threshold (“*determin[e] the decline in performance of said storage device*”) based on these logged communications. *See* [7], *supra*; Ex. C ¶ 120.

13. Claim 18

Claim [18] depends from claim 12 and recites “*program code for deriving a threshold value for the performance of said storage device, wherein said program code for responding to said decline in the performance of said storage device responds when the performance of said*

storage device satisfies said threshold value thereof.” Ex. A, claim 18. *Schibilla-Nagar* teaches this limitation for the reasons discussed in claim 1. *See* [1pre]-[1c], *supra*; Ex. C ¶ 121. For example, *Schibilla-Nagar* teaches “*deriving a threshold value for the performance of said storage device*” for the reasons in [1b]-[1c]. *See* [1b]-[1c], *supra*; Ex. C ¶ 122. And *Schibilla-Nagar* teaches “*wherein said program code for responding to said decline in the performance of said storage device responds when the performance of said storage device satisfies said threshold value thereof*” for the same reasons. *See* [1b]-[1c], *supra*; Ex. C ¶ 122.

14. Claim 19

Claim [19] depends from claim 12 and recites “*program code for reallocating data to another sector of said storage device.*” Ex. A, claim 19. *Schibilla-Nagar* teaches this limitation for the reasons discussed in [1c]. *See* [1c], *supra*; Ex. C ¶ 123.

15. Claim 20

Claim [20] depends from claim 19 and recites “*wherein said program code for reallocating reallocates at least same data on said storage device based on usage patterns of said a least some data.*” Ex. A, claim 20. *Schibilla-Nagar* teaches this limitation for the reasons discussed in [10]. *See* [10], *supra*; Ex. C ¶ 124.

16. Independent Claim 22

Schibilla-Nagar renders obvious independent claim 22. Ex. C ¶ 125.

a. [22pre]: “An apparatus for monitoring performance of a storage device”

Schibilla-Nagar teaches this preamble for the reasons discussed in [1pre]. *See* [1pre], *supra*; Ex. C ¶ 126.

- b. **[22a]: “means for evaluating communications between a computer system and said storage device to determine the performance of said storage device based at least in part on intercepted communications with said storage device”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1a]-[1c]. *See* [1a]-[1c], *supra*; Ex. C ¶ 127. For example, *Schibilla-Nagar* “*evaluat[es] communications between a computer system and said storage device*” by intercepting communications between a host and storage drive by monitoring those communications to determine the performance of the drive. *See* [1a]-[1c]; Ex. C ¶ 127. *Schibilla-Nagar* also teaches doing this via the claimed “*means*” (*see* § IV.E.1, *supra*), as both *Schibilla* and *Nagar* teach performing their processes via processors, hardware, firmware, and software. *See* [1a]-[1c], *supra*; Ex. C ¶ 127.

- c. **[22b]: “means for enhancing the performance of said storage device by reallocating at least some data on the storage device”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1c]. *See* [1c], *supra*; Ex. C ¶ 128. For example, *Schibilla* discloses “*enhancing the performance of said storage device by reallocating at least some data on the storage device,*” (*see, e.g.,* [1c], *supra*), and *Schibilla-Nagar* teaches doing so via the claimed “*means*” (*see* § IV.E.2, *supra*), as both *Schibilla* and *Nagar* teach performing their processes via processors, hardware, firmware, and software. *See* [1a]-[1c], *supra*; Ex. C ¶ 128.

17. Claim 23

Claim [23] depends from claim 22 and recites “*means for intercepting communications with said storage device.*” Ex. A, claim 23. *Schibilla-Nagar* teaches this limitation for the reasons discussed in [1a]. *See* [1a], *supra*; Ex. C ¶ 129. For example, *Nagar* teaches the claimed filter driver “*means*” (*see* § IV.E.3, *supra*), as *Nagar* is a textbook relating to filter drivers. *See* [1a], *supra*; Ex. C ¶ 129.

18. Independent Claim 25

Schibilla-Nagar renders obvious independent claim 25. Ex. C ¶ 130.

- a. **[25pre]: “A method for monitoring performance of a storage device”**

Schibilla-Nagar teaches this preamble for the reasons discussed in [1pre]. *See* [1pre], *supra*; Ex. C ¶ 131.

- b. **[25a]: “intercepting communications between a computer system and said storage device”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1a]. *See* [1a], *supra*; Ex. C ¶ 132.

- c. **[25b]: “analyzing said intercepted communications”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1b]. *See* [1b], *supra*; Ex. C ¶ 133.

- a. **[25c]: “reallocating at least some of said data on said storage device to enhance the performance of said storage device based on said analyzed communications”**

Schibilla-Nagar teaches this limitation for the reasons discussed in [1c]. *See* [1c], *supra*; Ex. C ¶ 134.

B. Ground 2: Claims 2, 3, 6, 14, 15, 24 and 26 Would Have Been Obvious over *Schibilla, Nagar, and Hospodor*

1. Overview of *Hospodor*

Hospodor describes techniques for measuring performance characteristics of a storage device. Ex. PA-3, Abstract; Ex. C ¶ 59. These metrics include various “access times” to perform operations on a disk and corrections to account for system overhead. *See, e.g.*, Ex. PA-3, Abstract 4:22-54; Ex. C ¶ 59. These corrections aim to ascertain a device’s direct performance while accounting for external parameters. *See, e.g.*, Ex. PA-3, 2:9-34, 4:22-54; Ex. C ¶ 59.

2. Claim 2

Claim [2] depends from claim 1 and recites “*measuring access time for said storage device.*” Ex. A, claim 2. *Schibilla-Nagar* in view of *Hospodor* teaches this additional limitation of claim 2. Ex. C ¶ 135.

Hospodor describes techniques for measuring “performance characteristics” of a disk drive, including “access time” (“*measuring access time*”). See, e.g., Ex. PA-3, Abstract, 1:7-26, 1:59-2:34, 5:25-42, 8:13-48; Ex. C ¶ 136. In one embodiment, *Hospodor* “measure[s] the amount of time required for the target to perform a specific task associated with its performance, such as the time required to access data.” Ex. PA-3, 3:64-4:4; see also, e.g., *id.*, 2:53-56 (describing “accurately determining the typical, average, and maximum access time characteristics of a target [storage] device”), 2:61-3:9, 8:49-59. *Hospodor* computes various performance measures, including “the total amount of time necessary to completely carry out a particular instruction” (t_{TOTAL}), a “host overhead time” (t_{HO}), and the time spent by the target drive “actually decoding and executing a particular command” (t_{ACCESS}), among others. See *id.*, 4:23-54; Ex. C ¶ 136; see also, e.g., Ex. PA-3, 5:25-42, 8:13-48, Figs. 5-6. *Hospodor* describes its total “access time” as a sum of several external ($t_{HOST OVERHEAD}$) and internal factors ($t_{TARGET OVERHEAD}$, $t_{SEEK POSITIONING}$, $t_{ROTATIONAL LATENCY}$, $t_{DATA TRANSFER}$). E.g., Ex. PA-3, 8:13-48. In one embodiment, for example, *Hospodor* relates total access time to overhead and internal access time as $t_{ACCESS} = t_{TOTAL} - t_{HOST OVERHEAD}$. See Ex. PA-3, 8:13-48; Ex. C ¶ 136. Because *Hospodor* measures various types of “access times,” including a total access time, a POSITA would have understood *Hospodor* to teach “*measuring access time for said storage device.*” Ex. C ¶ 136.

It would have been obvious to a POSITA to measure access time in combination with *Schibilla-Nagar*. Ex. C ¶ 137. *Hospodor* describes “access time” as a quintessential “performance characteristic[]” of a “storage device[].” See, e.g., Ex. PA-3, Abstract, 1:7-26,

1:59-2:8. *Schibilla*, in turn, uses any suitable “performance threshold” to conduct its data manipulation techniques. *See, e.g.*, Ex. PA-1, 6:36-61, 8:17-29. While *Schibilla* describes a number of read errors as one such performance metric (*see id.*)—a measure *Hospodor* similarly describes as “extremely important” (Ex. PA-3, 1:59-2:8)—*Hospodor* explains how accurate “access time” is another “crucial performance characteristic[]” (*id.*). Ex. C ¶ 137.

A POSITA in possession of *Schibilla-Nagar* would thus have been motivated to seek out other references, such as *Hospodor*, describing ways of using other crucial performance characteristics to extend *Schibilla*’s applicability and account for other performance metrics. Ex. C ¶ 138. Doing so would further *Schibilla*’s stated goal of preventing data loss and inefficiencies due to defective disks and poor data allocation by extending *Schibilla* to other performance parameters. *See, e.g.*, Ex. PA-1, 1:15-2:17; Ex. C ¶ 138. Indeed, *Hospodor*, a reference from the late 1980s, describes access times and read errors as important characteristics in rating storage device performance. *See* Ex. PA-3, 1:41-2:8; Ex. C ¶ 138. And as *Hospodor* explains, a POSITA would have appreciated the need to account for system overhead in access time calculations to derive a more accurate measure of true drive performance. *See* Ex. PA-3, 2:9-34; Ex. C ¶ 138.

A POSITA would have had a reasonable expectation of success in implementing *Hospodor*’s access time measurements into *Schibilla-Nagar*. Ex. C ¶ 139. As discussed, *Hospodor* explains how measuring access times for storage devices was a well-known concept in the 1980s, let alone at the time of the ’832 patent. Ex. C ¶ 139. Indeed, *Hospodor*’s description is replete with examples and equations explaining how to measure access times and account for system overhead to obtain more accurate measurements of drive performance. *See, e.g.*, Ex. PA-3, 2:9-34, 4:22-54, 5:25-62, 8:13-48. Measuring access times in *Schibilla-Nagar*—references dated roughly ten years later—would have thus amounted to nothing more than applying known

methods (e.g., access time measurements) to existing technology (e.g., *Schibilla*'s disk management routines and *Nagar*'s filter drivers) to achieve predictable results (e.g., measuring access times in *Schibilla-Nagar*). Ex. C ¶ 139.

3. Claim 3

Claim [3] depends from claim 2 and recites “*correcting said measured access time for system overhead.*” Ex. A, claim 3. *Schibilla-Nagar* in view of *Hospodor* teaches this additional limitation of claim 3 for the reasons discussed in [2]. *See* [2], *supra*; Ex. C ¶ 140.

For example, *Hospodor* describes subtracting a “host overhead” time (t_{HO} or $t_{HOST OVERHEAD}$) from “the total amount of time necessary to completely carry out a particular instruction” (t_{TOTAL}). Ex. PA-3, 4:22-54, 5:5-42, 7:43-63, 8:13-48; Ex. C ¶ 141. In certain aspects, *Hospodor* describes this host overhead as accounting for delays attributable to “the host system,” such as a CPU’s clock frequency and execution efficiency, rather than the storage device itself. *See, e.g.*, Ex. PA-3, 2:9-44, 4:22-54, 5:5-42, 8:31-48; Ex. C ¶ 141. *Hospodor* describes this latter measure as a more accurate indication of access time ascribable to the storage drive: $t_{ACCESS} = t_{TOTAL} - t_{HOST OVERHEAD}$. *See* Ex. PA-3, 4:22-54, 5:5-42, 7:43-63, 8:13-48; Ex. C ¶ 141. These disclosures mirror (and provide more detail than) the ’832 patent, which describes “*overhead*” as a value influenced by “external factors” influencing the “access time” of a storage drive. *See* Ex. A, 7:45-56. Because *Hospodor* accounts for various internal and external factors in its access time calculations, including external “host overhead,” a POSITA would have understood *Hospodor* to teach “*correcting said measured access time for system overhead.*” Ex. C ¶ 141.

A POSITA would have been motivated to correct for system overhead (as *Hospodor* discloses) to obtain a more accurate measure of the disk’s performance distinct from the host system. Ex. C ¶ 142. Indeed, *Hospodor* supplies this motivation itself. Ex. C ¶ 142. *Hospodor*

explains how “the rate at which different systems (or hosts) process information varies greatly” due to external factors, such as CPU speed and lengths of communication cables. *See* Ex. PA-3, 2:9-34. Thus, “the amount of time required to access data” will have “significant variations . . . even though an identical target is used during each test.” *Id.* Relatedly, this will cause devices of different manufacturers to report similar access times despite “the actual access times of each drive . . . be[ing] quite different.” *Id.*

Accordingly, a POSITA in possession of *Schibilla-Nagar* would have been motivated to correct for measured access times (as in *Hospodor*) to obtain a more accurate measure of drive performance distinct from external factors. Ex. C ¶ 143. A POSITA would have appreciated how more accurate time measures would improve *Schibilla*’s scoring metrics and would mitigate against prematurely identifying data sites as potentially defective. Ex. C ¶ 143. A POSITA would have understood these considerations would extend the life and efficiency of *Schibilla-Nagar*’s disk drive. Ex. C ¶ 143.

A POSITA would have had a reasonable expectation of success in correcting measured access times for system overhead. Ex. C ¶ 144. As discussed, these corrections involve nothing more than known measurements and simple mathematics. *See, e.g.*, Ex. PA-3, 4:22-54, 5:25-42, 8:31-48; Ex. C ¶ 144. Correcting measured access times would thus have amounted to applying known solutions (correcting access time for overhead) to existing technologies (e.g., *Schibilla*’s disk drives and *Hospodor*’s access times) to yield predictable results. Ex. C ¶ 144.

4. Claim 6

Claim [6] depends from claim 1 and recites “*determining an access location on said storage device and an access duration for data stored thereon, based on said intercepted communications.*” Ex. A, claim 6. *Schibilla-Nagar* and *Hospodor* teach this additional limitation of claim 6. Ex. C ¶ 145.

For example, *Schibilla-Nagar* teaches “*determining an access location on said storage device ... based on said intercepted communications*” for the same reasons discussed for claims 1 and 5 above. *See, e.g., [1a]-[1c], [5]; supra.* As for determining “*an access duration for data stored thereon, based on said intercepted communications,*” Patent Owner has taken the position that measuring “*access duration*” is taught by measuring “*access time*” as in claim 2. *See, e.g., Ex. L-3 at 24-26 (arguing that claim 6 is infringed by measurements of “access time”), 16 (citing the same evidence for claim 2).* While Requester does not accede to the propriety of this construction, *Hospodor* discloses this limitation under Patent Owner’s interpretation for the reasons discussed for claim 2 above. *See [2], supra.* Accordingly, the combination of *Schibilla-Nagar* and *Hospodor* teaches the subject matter of claim 6. Ex. C ¶ 146.

5. Claim 14

Claim [14] depends from claim 12 and recites “*program code for measuring access time for said storage device, and wherein said communications comprises at least said access time.*” Ex. A, claim 14. *Schibilla-Nagar* and *Hospodor* teach this additional limitation for the reasons discussed in claim 12 and [2]. *See [12pre]-[12b(c)], [2], supra; Ex. C ¶ 147.*

6. Claim 15

Claim [15] depends from claim 14 and recites “*program code*” for “*determining system overhead*” and “*correcting said access time for said system overhead.*” Ex. A, claim 15. *Schibilla-Nagar* and *Hospodor* teach this additional limitation for the reasons discussed in claim 12 and [2]-[3]. *See [12pre]-[12b(c)], [2], [3], supra; Ex. C ¶ 148.*

7. Claim 24

Claim [24] depends from claim 22 and recites “*means for measuring time to access said storage device, wherein said measured access time is evaluated to determine the performance of said storage device.*” Ex. A, claim 24. *Schibilla-Nagar* and *Hospodor* teach this additional

limitation, including the recited “*means*” (see § IV.E.4, *supra*) for the reasons discussed in claim 22 and [2]-[3]. See [22pre]-[22b], [2]-[3], *supra*; Ex. C ¶ 149.

8. Claim 26

Claim [26] depends from claim 25 and recites “*wherein intercepting said communications comprises determining access location, access frequency, and access duration for said data on said storage device.*” Ex. A, claim 26. *Schibilla-Nagar* and *Hospodor* teach this additional limitation for the reasons discussed in claim 22, 25, [2], [5], and [6]. See [22pre]-[22b], [25pre]-[25c], [2], [5], [6], *supra*; Ex. C ¶ 150. For example, *Schibilla-Nagar* teaches determining an “*access location*” and an “*access frequency*” for the reasons discussed for claim 5. See [5], *supra*. And *Hospodor* teaches determining an “*access duration*” under Patent Owner’s interpretation for the reasons discussed for claims 2 and 6. See [2], [6], *supra*; Ex. C ¶ 150.

C. Ground 3: Claims 8 and 18 Would Have Been Obvious over *Schibilla, Nagar, and Hetzler*

1. Overview of *Hetzler*

Hetzler describes methods for improving disk performance by monitoring read and write commands. Ex. PA-4, Abstract; Ex. C ¶ 60. *Hetzler* tracks various metrics of disk activity, such as access frequency, and uses dynamic thresholds to determine appropriate responses (e.g., placing the disk into a power-save mode). See Ex. PA-4, Abstract, 2:16-63, 9:32-46, 10:37-49, 11:38-12:2, Figs. 5-6; Ex. C ¶ 60.

2. Claim 8

Claim 8 depends from claim 7 and recites “*wherein analyzing said communications comprises deriving said threshold value based on said logged communications.*” Ex. A, claim 8. *Schibilla-Nagar* in view of *Hetzler* teaches this additional limitation of claim 8. Ex. C ¶ 151.

For example, *Hetzler* monitors a disk drive's workload by analyzing the read and write operations accessing the disk. *See* Ex. PA-4, Abstract, 2:16-63; Ex. C ¶ 152. In some embodiments, *Hetzler* improves the disk's energy performance by placing the disk into a power-saving mode depending on these access patterns. *See* Ex. PA-4, Abstract, 2:16-63; Ex. C ¶ 152. *Hetzler* selects the appropriate mode by comparing current access demands to a "previously calculated and continually updated threshold." Ex. PA-4, 2:16-63; Ex. C ¶ 152. The system updates these thresholds by adjusting factors "in response to tracking the actual performance of the system." Ex. PA-4, 2:16-63. These dynamic thresholds allow *Hetzler* to "adapt dynamically to varying workload situations." Ex. PA-4, 2:16-63; Ex. C ¶ 152.

Specifically, *Hetzler* updates various parameters in threshold calculations based on the recent access history of a given mode. *See id.*, 10:37-67, 11:38-12:2, 13:32-43; Ex. C ¶ 153. This enables the parameters to "dynamically" alter the thresholds based on past performance. Ex. PA-4, 13:32-43; Ex. C ¶ 153. To do so, the system tracks disk performance and adjusts these factors (*see* Ex. PA-4, 11:38-12:2), when the disk is performing poorly or avoiding power-save opportunities. *See id.*, 15:9-45; Ex. C ¶ 153. The former may occur when the system is using excess energy or has low response times. *See* Ex. PA-4, 16:5-17:48 ("energy/response penalties"). And the latter may occur when the system does not power-save when access frequency is low. *See id.*, 17:49-19:19 ("missed opportunity penalties"). In either case, the system uses these access frequency measurements to adjust factors impacting the threshold formula. *See* Ex. C ¶ 153; Ex. PA-4, 11:38-12:2, 19:20-21:34. This enables the system to "dynamically adjust[]" the parameters (and the threshold) "to follow changes in the access pattern." Ex. PA-4, 15:10-21; Ex. C ¶ 153.

Because *Hetzler* adjusts its thresholds “in response to tracking the actual performance of the system” (e.g., Ex. PA-4, Abstract, 2:16-63) a POSITA would have understood *Hetzler* to teach “*wherein analyzing said communications comprises deriving said threshold value based on said [] communications.*” Ex. C ¶ 154. And in combination with *Schibilla-Nagar*, a POSITA would have understood these access commands and accompanying information (e.g., *Hetzler*’s access times) would be “*logged.*” See, e.g., Ex. C ¶ 154; [1a]-[1c], [7], *supra*.

A POSITA would have been motivated to dynamically alter *Schibilla-Nagar*’s performance thresholds (as in *Hetzler*) for several reasons. Ex. C ¶ 155. First, a POSITA would have combined *Schibilla-Nagar* to achieve the energy performance benefits *Hetzler*’s power-save modes provide. E.g., Ex. C ¶ 155; Ex. PA-4, 2:10-63. *Schibilla* and *Hetzler* both describe ways of improving disk performance. Ex. C ¶ 155. A POSITA in possession of *Schibilla-Nagar* would thus have sought out additional ways of improving disk drive performance, like *Hetzler*, to account for other ways of improving an actual disk drive. Ex. C ¶ 155.

Second, a POSITA would have appreciated how dynamically adjusting *Schibilla-Nagar*’s performance thresholds would allow its disk drive to adapt to actual performance implementations. Ex. C ¶ 156. Indeed, *Hetzler* discusses this benefit. See, e.g., Ex. PA-4, Abstract, 1:61-65, 2:16-63; Ex. C ¶ 156. A POSITA would have appreciated how using other performance metrics, such as *Hetzler*’s access frequency, would expand *Schibilla-Nagar*’s capabilities and would benefit from dynamically changing thresholds. Ex. C ¶ 156. For example, a POSITA would have appreciated how *Schibilla*’s disk drive would benefit from adjusting its performance thresholds to avoid prematurely designating data sites as defective, to account for actual device performance, and to incorporate new and different performance metrics. Ex. C ¶ 156.

A POSITA would have had a reasonable expectation of success in combining *Hetzler* with *Schibilla-Nagar* to implement dynamically changing thresholds based on logged communications. Ex. C ¶ 157. As discussed, *Schibilla-Nagar* would have logged the access commands (e.g., read and write commands) used to compute *Hetzler*'s access frequency metrics. *See* Ex. C ¶ 157; [1a]-[1c], [7], *supra*; Ex. PA-4, Abstract. And *Hetzler* describes how performance thresholds may be predefined, user-defined, or dynamic. *See, e.g.*, Ex. PA-4, 13:31-14:45; Ex. C ¶ 157. Computing dynamic thresholds from logged communications, rather than simply relying on predefined numbers, would thus have merely applied known techniques (e.g., dynamic thresholds) to existing solutions (e.g., *Schibilla-Nagar*'s threshold-based performance drive) to achieve predictable results. Ex. C ¶ 157.

3. Claim 18

Claim [18] depends from claim 12 and recites similar subject matter as claims 1, 7, and 8. *See* [18], *supra*; Ex. C ¶ 158. Unlike claim 8, however, claim 18 does not require its “*threshold value*” to arise from “*logged communications*.” Accordingly, *Schibilla-Nagar* in view of *Hetzler* teaches this limitation for the reasons discussed in [8]. *See* [8], *supra*; Ex. C ¶ 158.

D. Ground 4: Claims 11 and 16 Would Have Been Obvious over *Schibilla, Nagar, and Jensen*

1. Overview of *Jensen*

Jensen discloses ways of defragmenting disk drives to increase their performance. *See, e.g.*, Ex. PA-5, Abstract, 2:40-48; Ex. C ¶ 61. *Jensen*'s techniques improve disk efficiency, particularly for operating systems such as Windows NT. *See, e.g.*, Ex. PA-5, 1:41-2:28, 2:40-48, 3:49-61, 5:22-44; Ex. C ¶ 61.

2. Claim 11

Claim [11] depends from claim 1 and recites “*wherein responding to said declining performance of said storage device further comprises defragmenting at least a portion of said storage device.*” Ex. A, claim 11. *Schibilla-Nagar* in view of *Jensen* teaches this additional limitation. Ex. C ¶ 159.

For example, *Schibilla-Nagar* teaches “*responding to said declining performance of said storage device*” for the reasons discussed in [1c]. *See* [1c], *supra*; Ex. C ¶ 160. As part of this process, *Schibilla* explains how it is “preferable” to store data together to keep it “contiguous.” *See, e.g.*, Ex. PA-1, 2:62-3:13, 7:23-47, 8:50-65, 10:15-41; Ex. C ¶ 160. *Schibilla* further discloses how it was known to keep data contiguous when assessing potentially defective data sites. *See* Ex. PA-1, 2:15-32. While *Schibilla* discloses the preference for keeping data contiguous (*see supra*), it does not expressly disclose “*defragmenting*” the drive after its conditional reassignment process.

Jensen discloses these details. Ex. C ¶ 161. Indeed, *Jensen* describes techniques for defragmenting disk drives in a computer system, particularly those running Windows NT. *See* Ex. PA-5, Abstract, 1:6-9, 2:32-48, 3:49-61. *Jensen*’s defragmentation process generally comprises identifying fragmented files, determining whether the disk contains sufficient free space, moving some of or the entire file to contiguous locations, and consolidating free space. *See* Ex. PA-5, 3:49-4:4, Figs. 4-9, Ex. C ¶ 161; *see also, e.g.*, Ex. PA-5, 8:24-9:30 (describing a partial and total defragmentation), 9:33-10:8 (describing moving files), 13:65-14:47. This process minimizes file fragmentation and the number of accesses required to transfer files. Ex. PA-5, 3:5-11; Ex. C ¶ 161. Because *Jensen* defragments some or all file content on a disk, a POSITA would have understood *Jensen* to disclose “*defragmenting at least a portion of said storage device.*” Ex. C ¶ 161.

A POSITA would have found it obvious to defragment a portion of a disk drive (as in *Jensen*) in response to detecting that a disk's performance has declined (as in *Schibilla-Nagar*). Ex. C ¶ 162. Throughout its disclosure, *Schibilla* discusses the importance of improving disk drive performance and discloses a preference for keeping data contiguous. *See, e.g.*, Ex. PA-1, 2:62-3:13, 6:36-61, 7:23-47, 8:17-29, 8:50-65, 10:15-41; Ex. C ¶ 162. A POSITA would have understood that *Schibilla*'s conditional reassignment process creates instances of noncontiguous data. *See* Ex. C ¶ 162; *see also, e.g.*, Ex. PA-1, 7:23-47, 8:50-65, 10:15-41 (explaining how it is “preferable” to keep data contiguous and reassigning data only when necessary). And as *Jensen* teaches, certain operating systems—such as Windows NT when using *Nagar*'s filter driver—commonly fragmented data. *See, e.g.*, Ex. PA-5, 1:41-2:28, 2:40-48, 3:49-61; Ex. PA-2, 33-36, Ex. C ¶ 162. It was thus well understood at the time of the '832 patent to defragment disk drives to keep data contiguous and increase performance. *See* Ex. C ¶ 162; Ex. PA-5, 2:40-48.

This would have led a POSITA in possession of *Schibilla-Nagar* to *Jensen*, whose disclosure addresses these issues by efficiently defragmenting drives for Windows NT platforms. Ex. C ¶ 163; Ex. PA-5, 2:40-48, 3:49-61, 5:22-44, 5:66-6:12. A POSITA would have appreciated how *Jensen*'s defragmentation process would fulfill *Schibilla-Nagar*'s goal of maintaining contiguous data in an efficient manner, especially for a Windows NT system. Ex. C ¶ 163; Ex. PA-5, 3:5-11. And like the reasons discussed for [3], a POSITA would have appreciated how defragmenting *Schibilla-Nagar*'s disk drive would improve its performance by reducing the instances of premature defective warnings. Ex. C ¶ 163. Thus, a POSITA would have found it obvious to defragment *Schibilla-Nagar*'s disk drive (as in *Jensen*) in response to detecting a decline in performance (as in *Schibilla-Nagar*). Ex. C ¶ 163. A POSITA would have understood that this defragmentation procedure may occur as part of *Schibilla-Nagar*'s conditional

reassignment procedure, after a certain number of conditional reassignments (e.g., after *Schibilla-Nagar*'s spare data pool became full or exceeded some threshold capacity), or other consideration to improve drive performance. Ex. C ¶ 163.

A POSITA would have had a reasonable expectation of success in defragmenting *Schibilla-Nagar*'s disk drive in response to a detected decline in performance. Ex. C ¶ 164. As discussed, *Schibilla-Nagar* already monitors disk performance and responds by potentially reallocating data on a disk. *See, e.g.* [1c], *supra*; Ex. C ¶ 164. Disk defragmentation was a ubiquitous process at the time of the '832 patent's earliest priority date. Ex. C ¶ 164. And here, *Jensen*'s defragmentation process uses the same Windows NT File System (NTFS) and File Allocation Table (FAT) described throughout *Nagar*'s filter driver discussion:

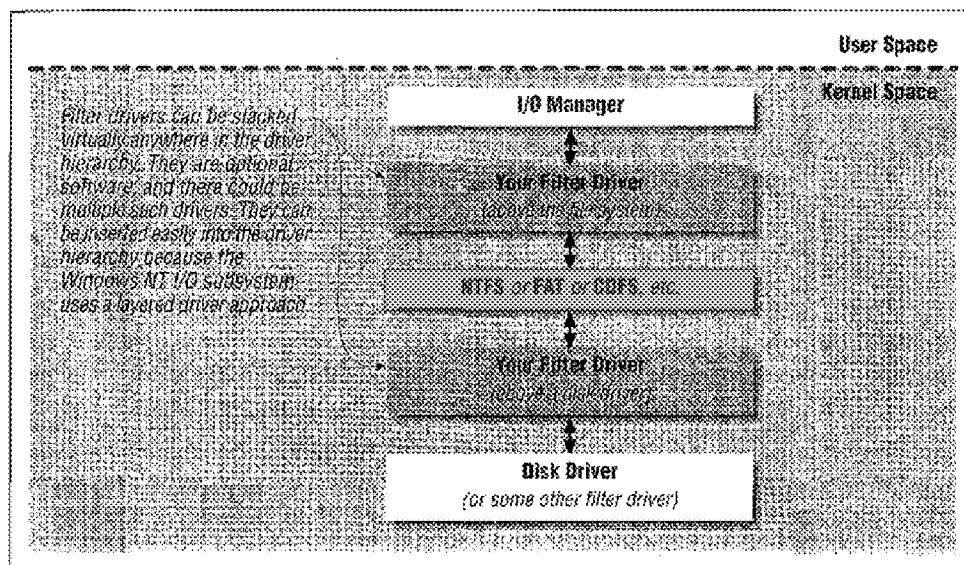


Figure 2-6. Filter drivers in the driver hierarchy

See, e.g., Ex. PA-2, 22-23, 33-35, 121, 258-259, 359-360; Ex. PA-5, 2:40-48, 3:49-61, 5:22-44, 5:66-6:23, 7:30-55, 8:6-23, Figs. 4-6; Ex. C ¶ 164. Thus, defragmenting some or all of *Schibilla-Nagar*'s disk drive would reflect nothing more than applying known processes (e.g., disk

defragmentation) to existing methods (e.g., *Schibilla-Nagar*'s disk monitoring and reallocation processes) to achieve predictable results. Ex. C ¶ 164.

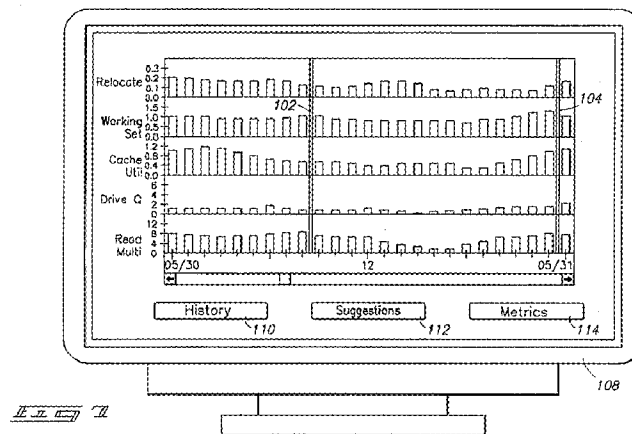
3. Claim 16

Claim 16 depends from claim 12 and recites “*program code for responding to said decline in the performance of said storage device further comprises defragmenting at least a portion of said storage device based on said communications.*” Ex. A, claim 16. *Schibilla-Nagar* and *Jensen* teach this additional limitation for the reasons discussed in claim 12 and [11]. See [12pre]-[12b(c)], [11], *supra*; Ex. C ¶ 165.

E. Ground 5: Claim 21 Would Have Been Obvious over *Schibilla, Nagar, and Voigt*

1. Overview of *Voigt*

Voigt describes techniques for improving data storage systems. *E.g.*, Ex. PA-6, Abstract; Ex. C ¶ 62. *Voigt* measures various performance metrics for a disk drive and graphically presents them to users:



Ex. PA-6, 4:64-5:13, 6:10-13, Fig. 7; Ex. C ¶ 62.

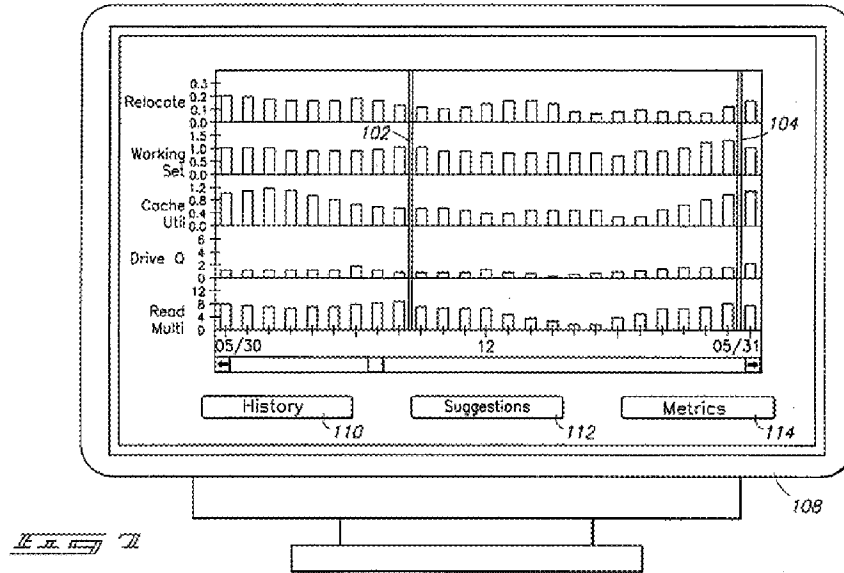
2. Claim 21

Claim 21 depends from claim 12 and recites “*a graphical user interface for reporting the performance of said storage device to a user.*” Ex. A, claim 3. *Schibilla-Nagar* in view of *Voigt* teaches this additional limitation. Ex. C ¶ 166.

As a preliminary matter, this limitation would have been obvious over *Schibilla-Nagar*. Ex. C ¶ 167. For example, *Schibilla* explains how it keeps a list of sites with grown defects “so that the user can retrieve the information at a later time.” Ex. PA-1, 10:42-55. *Nagar* likewise discloses how its filter driver “provide[s] adequate status reports to the system administrator,” such as “performance-related statistics” in the event of “error conditions.” Ex. PA-2, 66; Ex. C ¶ 167.

While *Schibilla* and *Nagar* do not expressly disclose reporting this information via a “graphical user interface” (GUI), a POSITA would have found this implicit in the references (let alone obvious) given the ubiquity of graphical interfaces to display information at the time of the ’832 patent. Ex. C ¶ 168. A POSITA would thus have found it obvious to display “performance related statistics” (as in *Nagar*) in the event of read error conditions (as in *Nagar* and *Schibilla*) to a user via a GUI (“*a graphical user interface for reporting the performance of said storage device to a user*”). Ex. C ¶ 168. A POSITA would have been motivated to present these statistics to, as *Nagar* instructs, “provide adequate status reports to the system administrator,” “allow[] the administrator to try to rectify the problem,” and “be aware that certain loss of functionality has either already occurred or might occur shortly.” Ex. PA-2, 66; Ex. C ¶ 168.

Nevertheless, to the extent Patent Owner disputes these teachings, *Voigt* teaches this very limitation. Ex. C ¶ 169. *Voigt* measures various “performance metrics,” such as blocks read, wait queues, and access times. Ex. PA-6, 4:64-5:13, 5:39-65. *Voigt* can display this information to the user via a graphical user interface:



See Ex. PA-6, 6:10-36 (“Preferably, the performance history is graphically displayed on visual display screen 18 in Fig. 1.”), 8:25-34, Fig. 7, claims 4-5, 11, 15; Ex. C ¶ 169. *Voigt* then enables users to interact with this display. Ex. PA-6, 2:6-20. A POSITA would have understood that *Voigt*’s “graphical[] display[]” of “performance history” reflects “a graphical user interface for reporting the performance of said storage device to a user.” Ex. C ¶ 169.

A POSITA would have been motivated to display *Schibilla*’s performance metrics and *Nagar*’s driver information via a GUI (as in *Voigt*). Ex. C ¶ 170. As discussed, both *Schibilla* and *Nagar* already disclose presenting information to a user regarding a disk drive’s performance history and error conditions. *See supra*. A POSITA would thus have been motivated to present this information to a user via a GUI to enable the user to view the information, interact with it, and take further action, as both *Nagar* and *Voigt* teach. *Id.*; see also Ex. PA-2, 66; Ex. PA-6, 6:10-36; Ex. C ¶ 170.

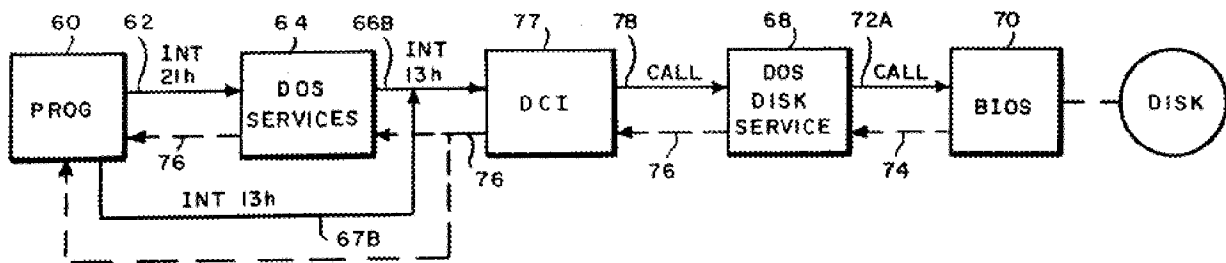
A POSITA would have had a reasonable expectation of success in displaying performance information, as displaying information in a GUI was ubiquitous at the time of the ’832 patent. Ex. C ¶ 171. Displaying performance information to the user would have merely

combined known elements (e.g., *Voigt*'s GUI) with existing technology (e.g., *Schibilla*'s disk monitoring, *Nagar*'s filter driver, disk performance measurements) to produce predictable results (displayed performance information in a GUI). Ex. C ¶ 171.

F. Ground 6: Claims 1-4, 6, 7, and 25 Would Have Been Obvious over *Wolf*

1. Overview of *Wolf*

Wolf describes techniques for monitoring a storage device's performance. *See* Ex. PA-14, Abstract. *Wolf* uses a software-based "disk-call interceptor" (DCI) to intercept data requests between a program and disk:



E.g., id., Abstract, Fig. 8; *see also id.*, 21:31-44 (explaining how the disk-call interceptor "intercepts requests to retrieve data from the [storage] medium"). The interceptor works in concert with a disk-read timer to measure the transfer times for data requests and an analysis program that examines the historical patterns for the targeted data location. *E.g., id.*, Abstract, 6:58-7:25, 17:15-55. When the routines detect a problem—such as read errors or transfer times exceeding an acceptable limit—the system takes remedial actions. *E.g., id.*, Abstract, 6:58-7:25, 20:24-41. These actions may include correcting identified errors or relocating affected data to a safe location on disk. *E.g., id.*, Abstract, 3:21-39, 6:58-7:11. The interceptor maintains a history of these events and measurements in a log file reflecting the historical performance of the data sector. *E.g., id.*, Abstract, 7:12-25; Ex. C ¶ 173.

2. Independent Claim 1

a. [1pre]: “A method for monitoring performance of a storage device”

To the extent the preamble is limiting, *Wolf* teaches the preamble. *Wolf* is directed towards a “[m]ethod and apparatus for use in a computer having peripheral data storage, to monitor data transfers to detect defective or marginally defective storage areas.” Ex. PA-14, Abstract. Indeed, *Wolf*’s purpose is to resolve issues in “being able to monitor the data storage performance of [a] medium in its normal operating environments.” *Id.*, 1:55-61. *Wolf*’s system uses routines that “monitor[] data retrieval from a peripheral storage medium” while the computer operates. *Id.*, 3:22-29. As discussed below, *Wolf* teaches the remaining elements of claim 1 involving monitoring performance of a storage device, thus teaching “*monitoring performance of a storage device.*” See [1a]-[1c], *infra*; Ex. C ¶ 176.

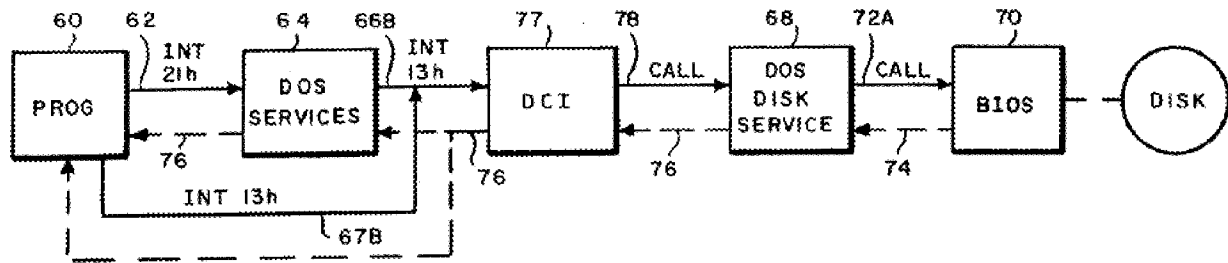
b. [1a]: “intercepting communications between a computer system and said storage device”

Wolf teaches this limitation. Ex. C ¶ 177. *Wolf* discloses a “disk-read timer” and “disk-call interceptor” to “gather information indicative of disk data” read problems and log that data into a log file. Ex. PA-14, 6:57-7:11. To do so, the interceptor “intercepts requests” from a computer program “to retrieve data from the [storage] medium,” thereby “*intercepting communications between a computer system and said storage device.*” *Id.*, 21:24-44; Ex. C ¶ 177.

Specifically, the interceptor resides “in the computer’s interrupt chain” and is “invoked during both the initiation and conclusion of storage operations.” Ex. PA-14, Abstract. When a program calls for data access, the disk-read timer interrupts the communication, computes data transfer measurements, and provides these measurements to the disk-call interceptor to evaluate system performance. *Id.*, 6:57-7:11; *see also id.*, 3:22-39. The interceptor “intercepts”

(“*intercept[s]*”) these data requests, analyzes the computed times, and determines whether an “error condition” occurred for the operation. *Id.*, 21:24-44; Ex. C ¶ 178.

The disk-call interceptor (DCI) and disk read timer (DRT) reside between a program (PROG) and disk (DISK), confirming *Wolf* “*intercept[s] communications between a computer system and said storage device*”:



Id., Fig. 8; Ex. C ¶ 179; *see also* Ex. PA-14, Figs. 3, 5 (disk-read timer interrupting PROG), 7 (showing how the interrupts occur between a program and disk), 9 (similar to Figure 8). The disk-call interceptor “needs to be in the disk-call chain in order to distinguish disk read operations from other disk operations before they start.” Ex. PA-14, 7:38-49. This enables the disk-call interceptor to intercept communications such that “whenever a program or the operating system thinks it is making a call to its low-level disk interface routine, it will actually be making a call to the disk-call interceptor.” *Id.*, 15:6-27; Ex. C ¶ 179.

For example, when a disk-read operation begins, the interceptor sets a “fast-tick flag” to alert the disk-read timer to measure time increments. Ex. PA-14, 10:33-46, 12:20-45. During this time, the interceptor logs events into a log file, and an analysis program analyzes these events to monitor system performance. *See, e.g., id.*, 16:58-17:3, 17:58-18:6; Ex. C ¶ 180. As explained below, the interceptor logs events that “indicate[] a media failure or potential media failure” and “temporarily relocates data” in defective regions to a different region. Ex. PA-14, 17:58-18:6; *see also* Ground 6, [1b]-[1c], *infra*.

Because *Wolf*'s disk-read timer and disk-call interceptor intercept and interrupt communications between a computer program and disk, a POSITA would have understood *Wolf* discloses “*intercepting communications between a computer system and said storage device.*” Ex. C ¶ 181.

To the extent Patent Owner contends such an interception step must occur via a filter driver—an interpretation to which Requester does not necessarily accede—a POSITA would have found it obvious to do so for the same reasons discussed above for *Schibilla-Nagar*. See, e.g., § VII.A.4.b, *supra*; Ex. C ¶ 182. For example, *Wolf* already describes implementing its disk-read timer as a “device driver” that “functions as an interface between the operating system and a peripheral device” (here, the disk). Ex. PA-14, 9:56-68. *Wolf* teaches doing the same for the disk-call interceptor. See *id.*, 15:41-65 (discussing “the disk-call interceptor [is] part of a device driver which in turn is part of the operating system”), 16:49-55 (stating the device driver “comprises the disk-read timer and the disk-call interceptor”). *Wolf* similarly discloses using its disk-call interceptor to “intercept[]” data transfers between a program and disk (*see supra*), and a POSITA would have known that filter drivers existed in the art to perform precisely this task (*see* § VII.A.4.b, *supra*). Indeed, *Wolf*'s disk-call interceptor performs similar functions as filter drivers known at the time of the '832 patent. Ex. C ¶ 182.

A POSITA would thus have been motivated to implement *Wolf*'s disk-call interception routines (as well as disk-read timer and analysis routines) as a filter driver to achieve the benefits known to inure to filter drivers, including (1) avoiding restrictions extant operating systems placed on direct disk manipulation, (2) streamlining development (e.g., permitting developers to update a single piece of software rather than various pieces of device-specific hardware), and (3) extending *Wolf*'s existing routines without having to write new file system drivers. See

§ VII.A.4.b, *supra*; Ex. C ¶ 183. Indeed, as discussed, *Wolf* teaches implementing its disk-read timer, disk-call interceptor, and other routines as intermediate drivers interfacing an operating system and a storage device. Ex. PA-14, 9:56-68, 15:6-27, 15:41-65. A POSITA would have had a reasonable expectation of success in implementing *Wolf*'s routines as a filter driver, as artisans had already implemented disk failure prediction processes as filter drivers at the time of the '832 patent. *See* Ex. C ¶ 75. Implementing *Wolf*'s routines as a filter driver would thus amount to nothing more than combining existing solutions to achieve predictable results. Ex. C ¶ 183.

c. [1b]: “analyzing said intercepted communications relative to a threshold value for the performance of said storage device”

Wolf teaches this limitation. Ex. C ¶ 184. *Wolf* “monitor[s] data transfers” to memory addresses and applies algorithms to determine whether an “error” has potentially occurred. Ex. PA-14, Abstract. These algorithms track various “*performance*” metrics, such as “time measurements of data retrievals” and indications of access errors (e.g., retries). *Id.*, 3:22-39; *see also, e.g., id.*, 6:57-7:11, 9:17-33 (“The disk-read timer measures the duration of data transfers by counting system timer ticks”). These errors may indicate “whether there are any areas of the storage medium that are unable to store data with integrity or are tending to lose the ability” to do so. *Id.*, 3:40-53. To conduct this analysis, an “analysis program” examines the log populated by the disk-call interceptor, maintains a history of recorded events, and applies algorithms to the historical data to perform “remedial measures” when the data indicates a failure mode. *Id.*, 7:13-37. These remedial measures may include temporarily relocating data on disk to a safe sector. *Id.*, 3:40-53; 6:57-7:11; Ex. C ¶ 184; *see also* Ground 6, [1c], *infra*.

Wolf's disk-call interceptor and analysis routines “*analyz[e]*” data transfers “*relative to a threshold value for the performance of the storage device.*” Ex. C ¶ 185. For example, *Wolf* explains how its disk-call interceptor “gathers the data transfer time measurements from the disk-

read timer” and “detects data transfers that took an excessive amount of time.” Ex. PA-14, 6:57-7:11; *see also id.*, 7:55-58 (discussing how the disk-read timer “measures the duration of data transfers during disk read operations”), 12:20-45, 20:24-41 (explaining how the disk-call interceptor compares transfers times “to a table of acceptable times”). These time measurements may “indicate that the data was not successfully read on the first attempt” and necessitated additional retries. *Id.*, 6:57-7:11. The disk-call interceptor may then “relocate” the impacted data “to a location accessible to the analysis program [for the] data which took an excessive time to read.” *Id.*; Ex. C ¶ 185; *see also* Ex. PA-14, 17:4-55 (discussing analysis of historical data and relocation).

A POSITA would have understood that identifying data transfers taking an “excessive amount of time” discloses a “*threshold value*.” Ex. C ¶ 186. The term “excessive” denotes the existence of one or more threshold values distinguishing excessive from nonexcessive times. *Id.* Times exceeding this threshold are “excessive,” while those below it are not. *Id.* Indeed, *Wolf* elsewhere discloses—and even claims—comparing measured transfer times to a “table of acceptable times” or an “acceptable time limit.” *See* Ex. PA-14, 20:24-41, 24:3-21, 28:1-9. A POSITA would have appreciated these “acceptable” times and “acceptable time limit” also reflect a “*threshold value*,” as transfers occurring within this “limit” are deemed “reasonable.” Ex. C ¶ 186; *see also* PA-14, 20:24-41, 24:3-21.

Wolf similarly describes analyzing a disk’s “historical data” to determine whether a disk area has become “defective to the point where [it] should be designated to the operating system and unusable” and its data relocated. Ex. PA-14, 17:4-14. To make this determination, *Wolf* compares a drive’s “historical data” to “criteri[a]” to known product specifications. *Id.*, 17:4-37; *see also, e.g.*, 3:22-39 (disk-call interceptor monitors for indications of error corrections). *Wolf*

devotes several claims to this concept. *See, e.g., id.*, 22:22-31 (claiming determining whether disk sectors are defective), 22:43-51, 23:16-25, 23:38-47, 24:61-25:3, 26:1-9, 26:21-31, 26:64-27:5, 27:17-26, 27:43-57. A POSITA would have appreciated how this defection point reflects a “*threshold value*,” as it represents the “point” (e.g., level or degree) where *Wolf* distinguishes defective from nondefective areas based on a “criterion.” Ex. C ¶ 187.

Because *Wolf* compares a disk’s historical performance against acceptable data transfer times, excessive time limits, and defective criterion points via intercepted communications, a POSITA would have understood *Wolf* discloses “*analyzing said intercepted communications relative to a threshold value for the performance of said storage device*.” Ex. C ¶ 188.

To the extent Patent Owner incorrectly disputes whether *Wolf*’s “excessive” time determinations, tables of “acceptable” times, “acceptable time limit,” and/or “point” of defect detection disclose a “*threshold value*,” a POSITA would have considered this obvious. Ex. C ¶ 189. A POSITA would have been motivated to use a simple threshold value as an example of an “excessive” or “acceptable time limit”—and would have found it obvious to try such an implementation—given that a single threshold value reflects the simplest way to distinguish unacceptable (“excessive”) from acceptable (“reasonable”) data transfer times. Ex. C ¶ 189; *see also* Ex. PA-14, 6:57-7:11, 20:24-41, 24:3-28 (claiming the use of a “table of acceptable time limits”), 28:10-15 (same). An ordinary artisan would likewise have found it obvious to use a threshold value as a “criterion” for the “point” at which *Wolf* considers a drive defective, as doing so was a common practice at the time. *See* Ex. C ¶ 189; *see also, e.g.,* Ex. PA-1, 6:35-7:10 (using a number of read-retries as a predefined performance threshold); PA-12, Abstract, ¶¶ [0029]-[0030], [0034]-[0039], [0041] (identifying a disk as defective when the number of data integrity errors reaches a “threshold” point), PA-13, 4:49-61. A POSITA would have had a reasonable

expectation of success in using a threshold value to identify defective areas and/or data transfers taking an excessive amount of time, as doing so would apply using a simple time or value comparison (measured value vs. threshold) to known solutions (*Wolf*) to achieve predictable results. Ex. C ¶ 189.

- d. **[1c]: “responding to a decline in the performance of said storage device based on said analyzed intercepted communications by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device”**

Wolf discloses this limitation. Ex. C ¶ 190. When *Wolf* detects a performance decline—such as “excessive” access times or defective disk areas (*see* [1b], *supra*)—it takes “remedial action” including “relocating the [affected] data to a usable area.” Ex. PA-14, Abstract; *see also id.*, 3:40-53, 6:57-7:11, 15:41-65, 16:58-17:38, 17:58-18:6, 22:22-31. As discussed above, for example, *Wolf* detects “a decline in the performance of said storage device based on said analyzed intercepted communications” when it detects data transfers taking “an excessive amount of time,” transfers exceeding an “acceptable time limit,” or “defective or marginally defective storage area[s]” on disk. *Id.*, Abstract; *see also id.*, 6:57-7:25; 20:24-41, 24:1-28; Ex. C ¶ 190; Ground 6, [1b], *supra*.

Wolf also “respond[s]” to this drop in performance “by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device.” Ex. C ¶ 191. Upon detecting excessively slow data transfers or marginally defective storage areas, the disk-call interceptor or analysis routines “take[] remedial action” such as “relocating the data to a usable area.” Ex. PA-14, Abstract. *Wolf* discusses this relocation process throughout its disclosure. *See id.*, Abstract, 3:40-53, 6:57-7:11 (“The disk-call interceptor may temporarily relocate to a location accessible to the analysis program data which took an excessive amount of time to read.”), 7:12-25 (“Remedial action can include moving to a good area the data [stored in a defective

sector].”), 15:41-65, 16:58-17:38 (“The analysis program determines on the basis of the historical data whether any disk areas are defective to the point where they should be designated to the operating system as unusable, and the data relocated.”), 17:58-18:6. *Wolf* also dedicates several claims to the relocation concept. *See id.*, 22:22-31, 22:43-52, 23:16-25, 23:38-47, 24:62-25:3, 25:67-26:9, 26:21-31, 27:17-26, 28:48-57; Ex. C ¶ 191.

Finally, a POSITA would have understood *Wolf*’s relocation process “*enhance[s] continued operation of the storage device.*” Ex. C ¶ 192. *Wolf*’s drive continues to operate after data relocation. *See* Ex. PA-14, 3:40-53, 17:15-38. For example, relocating the data to another “area”—as opposed to a different drive—confirms *Wolf*’s disk continues normal operation. *Id.*, 3:40-53, 17:15-38. *Wolf* even describes procedures for remediating areas detected as defective. When *Wolf*’s analysis program detects a data error, it checks the area’s historical data to determine whether the error is intermittent or isolated. *Id.*, 17:15-38. If the area “has no history of errors,” the analysis program considers the area still “good” and may return data to that location. *Id.*; *see also id.*, 6:57-7:11 (describing a disk-call interceptor’s relocation as potentially “temporar[y]”), 15:41-65 (same), 24:29-39; Ex. C ¶ 192.

Thus, because *Wolf* relocates data to a safe area on disk in response to detecting defective areas or slow data transfer times, a POSITA would have appreciated *Wolf* discloses “*responding to a decline in the performance of said storage device based on said analyzed intercepted communications by automatically reallocating at least some data on said storage device to enhance continued operation of the storage device.*” Ex. C ¶ 193.

3. Claim 2

Claim [2] depends from claim 1 and recites “*measuring access time for said storage device.*” Ex. A, claim 2. *Wolf* discloses this additional limitation of claim 2. Ex. C ¶ 194.

As discussed above for claim 1, for example, *Wolf*'s disk-read timer “measure[s] the time duration of data transfer during disk read operations” to accomplish *Wolf*'s monitoring tasks. *See* [1pre]-[1c]; *see also* Ex. PA-14, 6:57-7:11. *Wolf* discusses measuring data retrieval times throughout its disclosure. *See, e.g.*, Abstract (measuring times for “data access”), 2:8-61 (discussing measuring data access times in the prior art), 3:22-39 (“The monitoring includes, without limitation, making time measurements of data retrievals”), 6:57-7:11, 7:55-58, 9:17-55 (“The disk-read timer measures the duration of data transfers by counting system timer ticks”), 12:20-45, 12:58-13:8, 20:24-41, 21:31-44 (measuring “an elapsed time of the operation performed for retrieving the data from [a] rotating storage medium”), 24:3-11, 25:15-24, 28:1-19. A POSITA would have appreciated that *Wolf*'s measurements for disk read and data retrieval times reflects “*measuring access time*.” Ex. C ¶ 195; *see also, e.g.*, Ex. A, 6:62-7:13 (referring to access time as the amount of time necessary to successfully access data on disk).

4. Claim 3

Claim [3] depends from claim 2 and recites “*correcting said measured access time for system overhead*.” Ex. A, claim 3. *Wolf* discloses this additional limitation of claim 3. Ex. C ¶ 196.

For example, *Wolf* explains how its system eliminates problems with existing access time calculations, including those due to “controller overhead.” Ex. PA-14, 20:42-50; *see also id.*, 12:46-68 (explaining how its system eliminates complications due to “seek and rotational latency”). To do so, *Wolf* places its disk-read time “at the ‘head’ of a system timer interrupt chain and maintain that position against other programs that may try to supplant it.” *Id.*, 6:58-7:11; *see also id.*, 9:34-55, 10:33-46 (explaining how the disk-read timer “takes steps to remain in the first place in the timer interrupt chain” when it is not counting time). When a disk-read operation begins, the disk-call interceptor sets a fast-tick flag to true, which instructs the disk-read timer to count time. *Id.*, 12:20-45. Once the disk-read timer detects this flag is false, it stops counting ticks and

passes timing information to the disk-call interceptor. *Id.*, 12:46-68. The disk-call interceptor then “arrive[s] at an adjusted tick count” by “subtract[ing] the tick count up to the change in the contents of the first address of the memory buffer from the total tick count.” *Id.* This adjustment “eliminate[s] any seek and rotational latency time preceding the actual transfer of data.” *Id.* *Wolf* further explains how it can retest data transfer times to account for inaccurate measurements due to “controller overhead.” *Id.*, 20:42-50. With these adjustments, *Wolf* “make[s] time measurements of data retrievals much more accurate[] than conventional test programs,” which did not account for items such as “seek latency time.” *Id.* 2:8-61; Ex. C ¶ 197; *see also* Ex. PA-14, 5:43-62.

5. Claim 4

Claim [4] depends from claim 1 and recites “*wherein intercepting said communications comprises intercepting an error reported by said storage device.*” Ex. A, claim 4. *Wolf* discloses this additional limitation of claim 4. Ex. C ¶ 198.

For example, *Wolf* teaches how conventional disk controllers provide an “error correction code (ECC)” as part of their “end-of operation status information.” *See* Ex. PA-14, 2:62-3:19, 18:7-25. In *Wolf*’s system, the disk-call interceptor intercepts this information by being “interposed between the disk interface routine and other programs.” *Id.*, 18:26-39. This placement ensures the disk-call interceptor receives access to such status information by the end of the disk operations. *Id.* The disk-call interceptor further “check[s] for indications that an error correction has taken place” and stores these indications in a log file. *Id.*, Abstract, 3:22-39; *see also id.*, 6:65-67, 7:12-38, 15:41-65, 25:46-55, 28:37-36. When *Wolf*’s analysis program detects new errors have occurred in the log file, it performs a procedure to correct potentially defective data sites. *Id.*, 17:15-38. *Wolf* recites this feature in several claims. *See, e.g., id.*, 22:44-52, 23:17-25, 23:39-47, 26:1-9, 26:23-31, 26:66-27:5, 27:19-26, 28:50-57. *Wolf* then analyzes the historical occurrence of these

corrections to determine potential causes, such as read/write faults or spindle bearing failures. *Id.*, 17:39-55; Ex. C ¶ 199.

6. Claim 6

Claim [6] depends from claim 1 and recites “*determining an access location on said storage device and an access duration for data stored thereon, based on said intercepted communications.*” Ex. A, claim 6. *Wolf* discloses this additional limitation of claim 6. Ex. C ¶ 200.

Wolf discloses “*determining an access location on said storage device ... based on said intercepted communications*” for the reasons discussed above. *See, e.g.*, Ground 6, [1b]-[1c], *supra*; Ex. C ¶ 201. For example, *Wolf* monitors data transfer times and errors occurring in “defective or marginally defective storage areas,” marks these areas as defective, and “relocat[es] the data to a usable area.” Ex. PA-14, Abstract. *Wolf* records this information in a log to create “a history of the [storage] medium per unit area, e.g., per sector” (e.g., a “*location*”). *Id.*, 3:40-53. This log reflects the historical access performance of a particular sector, area, or location on disk. *See id.*, 7:12-25 (maintaining and analyzing a “history of such events sorted by disk unit areas, such as sectors”), 12:20-45 (explaining how the disk-read timer inspects specified “memory addresses” for changes when counting time), 12:46-68 (same), 13:1-8, 13:49-52 (saving the “address” of an instruction), 14:46-56, 15:51-65, 17:15-38 (discussing how the analysis program attempts to “correct bad locations” on disk based errors detected at specification locations), 20:24-41 (logging read times for sectors and blocks of sectors). Ex. C ¶ 201. *Wolf* uses this information for various purposes, such as maintaining histories of “the location at which [an] error occurred.” Ex. PA-14, 17:23-55. *Wolf* recites this disclosure across several claims. *See id.*, 22:11-52 (reciting “logging information concerning each operation for retrieving data during which an error condition occurred, the information including the location on the medium from which data was retrieved”), 23:5-47, 24:28-39, 24:51-25:24, 25:57-26:31, 26:53-27:26, 28:16-58; Ex. C ¶ 201.

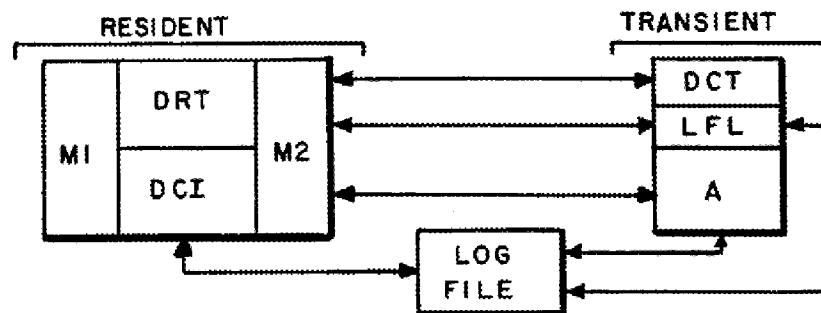
As for determining “an access duration for data stored thereon, based on said intercepted communications,” Patent Owner has taken the position that measuring “access duration” is taught by measuring access time as in claim 2. *See, e.g.*, Ex. L-3 at 24-26 (arguing that claim 6 is infringed by measurements of “access time”), 16 (citing the same evidence for claim 2). While Requester does not accede to the propriety of this construction, *Wolf* discloses this limitation under Patent Owner’s interpretation for the reasons discussed for claim 2 above. *See* [2], *supra*. For example, *Wolf* discloses measuring a “time duration of data transfer[s] during disk read operations.” Ex. PA-14, 6:58-7:11; *see also id.*, 7:55-58 (explaining how the disk-read timer “measures [] the duration of data transfers during disk read operations”), 9:17-33; Ex. C ¶ 202.

Accordingly, *Wolf* discloses “determining an access location on said storage device and an access duration for data stored thereon, based on said intercepted communications.” Ex. C ¶ 203.

7. Claim 7

Claim [7] depends from claim 1 and recites “logging said communications over time.” Ex. A, claim 7. *Wolf* discloses this additional limitation of claim 7. Ex. C ¶ 204.

As discussed above, for example, *Wolf*’s disk-call interceptor logs data transfer times and error events in a “log file”:



See Ex. PA-14, Fig. 3; *see also id.*, Abstract, 3:22-58 (explaining how *Wolf* “logs” uncorrectable errors, corrected errors, retries, and other information “into a file for later analysis”), 6:58-7:11

(“A primary function of the disk-call interceptor is to gather information indicative of disk data read problems and to log that information onto the log file.”), 12:58-13:8 (logging whether memory contents change), 15:41-65, 17:58-18:6, 20:24-41 (logging read operations); Ex. C ¶ 205. *Wolf*’s analysis program uses this log file to build a history of events on a per-sector basis and analyzes these events to fix detected problems. Ex. PA-14, 7:12-25; *see also id.*, 16:58-17:3 (referring to the log as “a continuously updated history of the storage media” on a per-sector basis), 17:23-55. *Wolf* devotes several claims to this logging concept. *See, e.g., id.*, 22:11-52 (reciting for example, “logging information concerning each operation for retrieving data during which an error condition occurred, the information including the location on the medium from which data was retrieved”), 23:5-47, 24:51-25:3, 25:57-26:31, 26:53-27:26, 28:37-58; Ex. C ¶ 205.

8. Independent Claim 25

Wolf teaches independent claim 25. Ex. C ¶ 206.

a. **[25pre]: “A method for monitoring performance of a storage device”**

Wolf teaches this preamble for the reasons discussed in [1pre]. *See* Ground 6, [1pre], *supra*; Ex. C ¶ 207.

b. **[25a]: “intercepting communications between a computer system and said storage device”**

Wolf teaches this limitation for the reasons discussed in [1a]. *See* Ground 6, [1a], *supra*; Ex. C ¶ 208.

c. **[25b]: “analyzing said intercepted communications”**

Wolf teaches this limitation for the reasons discussed in [1b]. *See* Ground 6, [1b], *supra*; Ex. C ¶ 209.

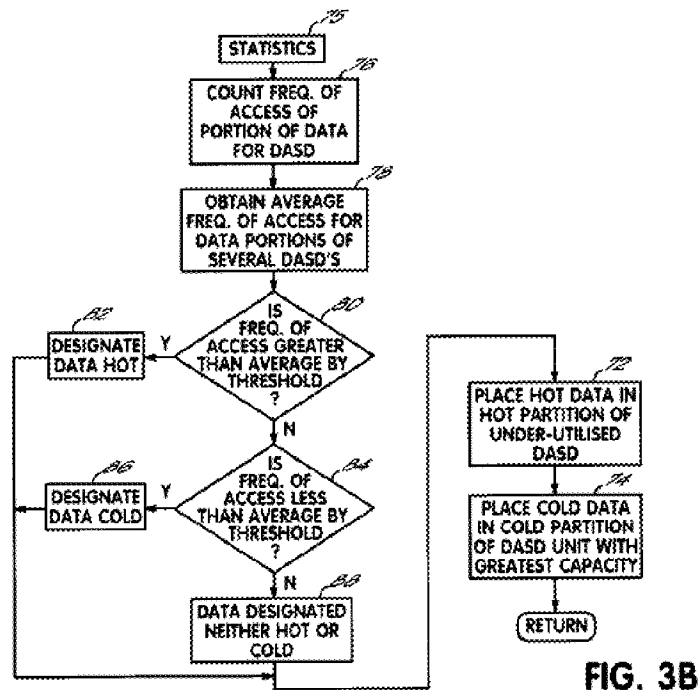
- d. [25c]: “reallocating at least some of said data on said storage device to enhance the performance of said storage device based on said analyzed communications”

Wolf teaches this limitation for the reasons discussed in [1c]. *See* Ground 6, [1c], *supra*; Ex. C ¶ 210.

G. Ground 7: Claims 5, 8, 10, and 26 Would Have Been Obvious over *Wolf* and *Christenson*

1. Overview of *Christenson*

Christenson is directed to techniques for improving a disk’s performance by dynamically reallocating data based on its frequency of use. *See* Ex. PA-15, Abstract. The system measures access frequencies for data on a DASD (direct access storage device) and designates that data as “HOT” (frequently accessed), “COLD” (infrequently accessed), or neither based on thresholds:



Id., Abstract, 8:34-9:20, Fig. 3B; Ex. C ¶ 174. *Christenson* can also rate the disks themselves as HOT, COLD, or neither. *See, e.g.*, Ex. PA-15, 3:21-45, 10:14-53, Fig. 5. To improve drive performance, *Christenson* dynamically moves HOT and COLD data to corresponding HOT and

COLD partitions on disks that are underutilized. *See, e.g., id.*, 9:26-54, 11:1-47, Fig. 6. This reallocation scheme reduces bottlenecks and provides faster data access. *See, e.g.,* Ex. PA-15, 8:34-9:20; Ex. C ¶ 174.

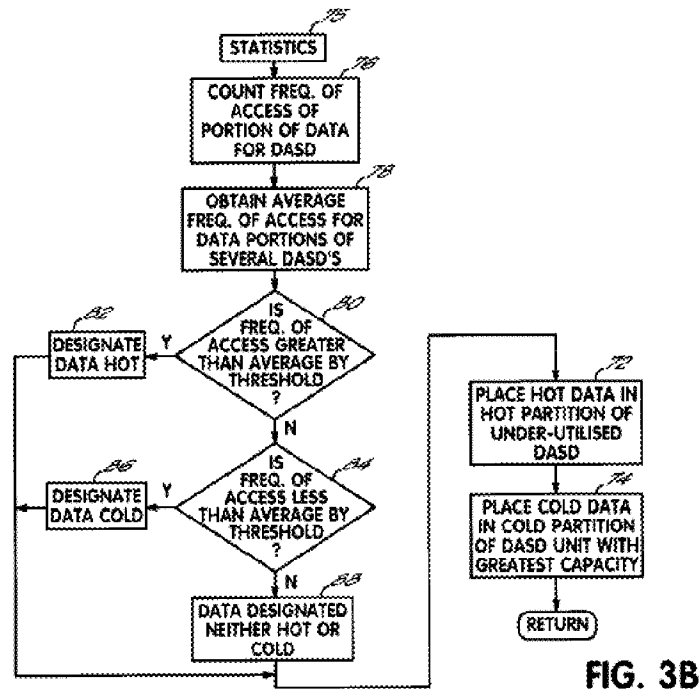
2. Claim 5

Claim [5] depends from claim 1 and recites “*determining an access location on said storage device and an access frequency for data stored thereon, based on said intercepted communications.*” Ex. A, claim 5. *Wolf* and *Christenson* teach this additional limitation of claim 5. Ex. C ¶ 211.

For example, *Wolf* discloses “*determining an access location on said storage device ... based on said intercepted communications*” for the reasons discussed above for claims 1 and 6. *See* Ground 6, [1pre]-[1c], [6], *supra*; Ex. C ¶ 212.

Christenson likewise discloses “*determining an access location on said storage device and an access frequency for data stored thereon.*” Ex. C ¶ 213. For example, *Christenson* monitors “portions of data on a plurality of DASD [direct access storage devices] units” (“*an access location*”) “and the number of times the portions of data on the DASD units are accessed within a given time period” (“*an access frequency for data stored thereon*”). *See* Ex. PA-15, 2:51-62; Ex. C ¶ 213; *see also* Ex. PA-15, Abstract. Given this “access frequency of the data,” *Christenson* discloses, “the data will be moved and otherwise managed to reduce system bottlenecks.” Ex. PA-15, 2:51-62. *Christenson* designates the “more frequently accessed” data as “HOT” and the “less frequently accessed data portions” as “COLD.” *Id.*; *see also, e.g., id.*, Abstract, 7:57-8:4, 10:15-53. In this way, *Christenson* “determines the frequency of access of the data on a DASD unit or among DASD units and moves the data on the DASD unit or among DASD units based on the data’s access frequency.” *Id.*, 6:46-7:10; Ex. C ¶ 213.

To make HOT and COLD designations, *Christenson* determines an “average frequency of access for data in the system.” Ex. PA-15, 3:21-45; *see also id.*, 8:12-33. Data accessed more than a threshold amount above this average is considered HOT, while data accessed less than a threshold amount below this average is COLD (data within this range is considered neutral):



E.g., id., 3:21-45, Fig. 3B; *see also id.*, 8:12-9:20 (computing average access frequencies and comparing access frequencies for particular data portions to this average); Ex. C ¶ 214.

Christenson moves HOT and COLD data among various drives to balance utilization (*e.g.*, Ex. PA-15, 2:63-3:8, 14:57-15:11), or within a HOT and COLD partition on a single drive (*e.g.*, *id.*, 3:9-20). *See also, e.g., id.*, 9:26-54 (discussing partitioning a drive with HOT and COLD portions where corresponding data is stored), 9:57-10:54 (determining drive utilization), 10:56-11:16 (discussing moving data between HOT and COLD partitions), Fig. 3B; Ex. C ¶ 214.

Because *Christenson* reallocates data based on determined access frequencies for specific data residing on a location on disk, a POSITA would have understood *Christenson* discloses

“determining an access location on said storage device and an access frequency for data stored thereon.” Ex. C ¶ 215.

A POSITA would have been motivated to determine data access frequencies, as in *Christenson*, within the disk monitoring system discussed in *Wolf*. Ex. C ¶ 216. As explained, *Wolf* already teaches monitoring a drive’s performance by continually measuring and relocating data based on logged access times. *See* [1pre]-[1c], [2]-[3], *supra*. A POSITA would have been motivated to further relocate this data based on access frequency, as taught in *Christenson*, to expand *Wolf*’s functionality, vary the parameters used to assess a data sector’s “historical” performance (Ex. PA-14, 3:40-53), and improve drive performance by reducing access times. Ex. C ¶ 216. Indeed, *Christenson* explains how a drive’s performance hinges in part on considerations such as the drive utilization, data contiguity, and data classifications. *See* Ex. PA-15, 1:50-2:17, 8:34-9:20; Ex. C ¶ 216. And as discussed above, a POSITA would have known the efficiency benefits that *Christenson*’s storage management system would provide to a disk drive. *See, e.g.*, Ex. PA-2, 260-261 (discussing HSM), 620-622; § VII.A.6, *supra*; Ex. C ¶ 216.

A POSITA would thus have been motivated to implement *Christenson*’s access frequency reallocation routines—including computing access frequencies—to improve *Wolf*’s device access performance and create a more detailed “history” used to assess the performance of the drive. *See* Ex. PA-14, 3:40-53, 7:12-25, 16:58-17:38; Ex. C ¶ 217. Doing so would store data more efficiently, improve access times, and reduce potential “thrashing” (moving data too often) due to poor access times. *See, e.g.*, Ex. PA-15, 6:46-7:10, 8:12-33, 10:14-53; Ex. C ¶ 217. For example, where *Wolf* teaches relocating data with slow access times to another disk location (*see, e.g.*, Ex. PA-14, 6:58-7:11), *Christenson* provides further details as to *why* this data may have slow access times (*e.g.*, higher access frequencies for HOT data) and *where* to move it (*e.g.*,

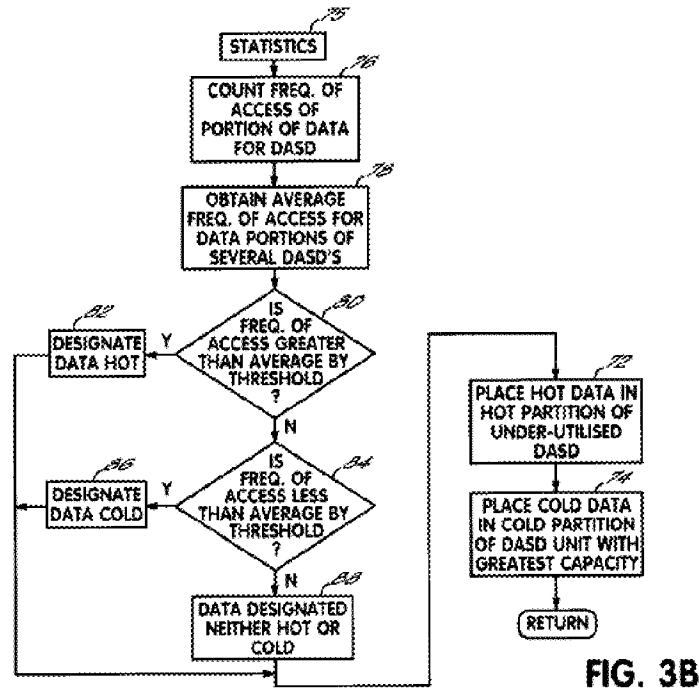
a HOT partition) (*see, e.g.*, Ex. PA-15, 8:34-9:20, 9:27-54, 10:56-11:47). Ex. C ¶ 217. A POSITA in possession of *Wolf* would thus have sought out references like *Christenson* in search of these solutions. Ex. C ¶ 217.

A POSITA would have had a reasonable expectation of success in computing *Christenson*'s access frequencies, as doing so would involve simple mathematical calculations involving "the number of times the portions of data ... are accessed within a given time period." Ex. PA-15, 2:51-62; Ex. C ¶ 218. Because *Wolf* and *Christenson* already monitor all such data accesses (*see, e.g.*, [1pre]-[1c], *supra*), a POSITA would have found it trivial to determine the number of times these accesses occurred during a given time span (as taught in *Christenson*). Ex. C ¶ 218. Indeed, as discussed, such frequency calculations were a well-known phenomenon in HSM. *See, e.g.*, Ex. PA-2, 260-261, 620-622; Ex-PA-15, 2:51-62; § VII.A.6, *supra*; Ex. C ¶ 218. Accordingly, computing access frequencies for *Wolf*'s system would have involved nothing more than combining existing solutions (HSM, access frequencies) with known technology (*Wolf*'s disk performance tracker) to achieve predictable results. Ex. C ¶ 218.

3. Claim 8

Claim 8 depends from claim 7 and recites "*wherein analyzing said communications comprises deriving said threshold value based on said logged communications.*" Ex. A, claim 8. *Wolf* and *Christenson* teach this additional limitation of claim 8. Ex. C ¶ 219.

As discussed above, *Christenson* decides whether to reallocate data based on how its access frequency compares to certain thresholds above and below an "average frequency of access." *See, e.g.*, Ex. PA-15, 3:21-45, 8:12-33, Fig. 3B; *see also* [5], *supra*. When data has an access frequency that exceeds the average by some "identification threshold," the system designates the data as HOT. Ex. PA-15, 8:34-9:20. When the access frequency of the data is less than another "identification threshold" below the average, the system considers the data COLD:



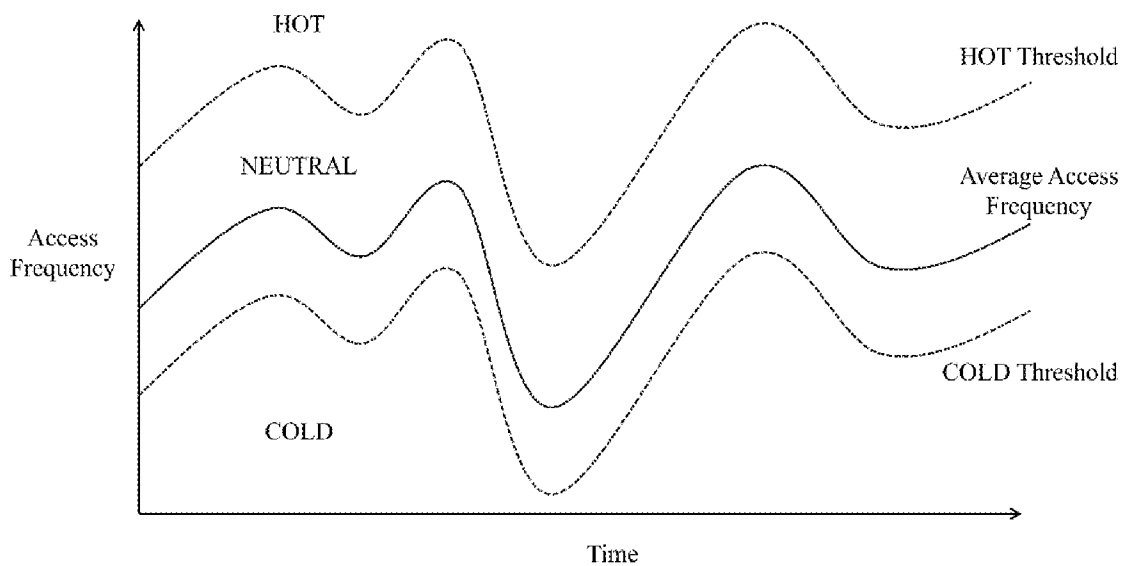
Id.; see also *id.*, Fig. 3B; Ex. C ¶ 220. These thresholds may be different (Ex. PA-15, 8:34-9:20), application-dependent (*id.*, 8:12-33), defined by a user (*id.*), hard coded (*id.*), or set upon installation for an entire system (*id.*). *Christenson* then reallocates data across various drives, or within HOT and COLD partitions of a single drive, based on these designations. See, e.g., *id.*, 8:34-9:20; see also, e.g., 3:9-20, 7:13-43, 9:26-53; Ex. C ¶ 220.

A POSITA would have understood that *Christenson*'s HOT and COLD thresholds⁴ are “derived ... based on said logged communications.” Ex. C ¶ 221. As discussed, *Christenson* monitors data access requests to drives, computes an average access frequency for data, and then reallocates data based whether a specific datum's access frequency exceeds or falls below this average beyond some threshold amount. See *supra*; see also [5], *supra*. A POSITA would have appreciated how *Christenson*'s HOT and COLD access frequency thresholds are “based on said

⁴ By “HOT” and “COLD” thresholds, Requester refers to the thresholds at which *Christenson* designates data as HOT or COLD, not the incremental delta between the average access time at those limits.

logged communications,” as they depend in part on the “average frequency of access” for all data in the system. *See* Ex. PA-15, 9:57-10:13; Ex. C ¶ 221. Data accessed above an upper threshold—which depends on an average access frequency—is HOT, while data access below a lower threshold—which also depends on the average access frequency—is COLD. *E.g.*, Ex. PA-15, 8:12-9:20; Ex. C ¶ 221.

A POSITA would also have appreciated how these HOT and COLD thresholds may vary across time. Ex. C ¶ 222. For example, as users continue to access data, create new data, etc., a POSITA would have understood how these operations impact the average access frequency for all data on the system as a whole:



Ex. C ¶ 222. As illustrated above, the HOT and COLD thresholds depend on the average access frequency and will thus dynamically evolve over time. Ex. C ¶ 222. Indeed, *Christenson* describes its system as “dynamic” to “dynamically manage data.” *See, e.g.*, Ex. PA-15, Title, 2:44-62, 11:17-47, 13:9-32, 17:7-30, 17:44-59; Ex. C ¶ 222. The data accesses and frequencies used to determine these thresholds and the HOT or COLD designations would be logged by *Wolf-Christenson*. *Id.*; *see also, e.g.*, Ground 6, [7], *supra* (explaining how *Wolf* logs data accesses and access times);

Ex. PA-15, 9:57-10:13 (counting “the frequency of access for all the data” for a given time period). Accordingly, a POSITA would have appreciated how *Christenson*’s HOT and COLD designations involve “*deriving said threshold value based on said logged communications.*” Ex. C ¶ 222.

A POSITA would have been motivated to implement *Christenson*’s dynamic, frequency-based thresholding within *Wolf* for the reasons as discussed for claim 5. *See* [5], *supra*; Ex. C ¶ 223. For example, a POSITA would have been motivated to log data accesses (as in *Wolf* and *Christenson*) and compute *Christenson*’s HOT and COLD thresholding to implement *Christenson*’s data reallocation procedure. *See* [5], *supra*; Ex. C ¶ 223. As discussed above, a POSITA would have been motivated to do so to expand the historical performance parameters examined by *Wolf*’s analyzer and further improve the performance of *Wolf*’s drive. Ex. C ¶ 223. Again, it was well known in the art that HSM, which reallocates data based on usage histories, provided performance benefits to disk drives. *See* [5], *supra*; Ex. C ¶ 223. To this end, *Christenson* expressly teaches how frequent requests for data would create data bottlenecks in speed and efficiency. *See, e.g.*, Ex. PA-15, 1:50-2:5, 8:34-9:20. Thus, a POSITA would have been motivated to implement *Christenson*’s HOT and COLD thresholds (which are based on logged communications and their counts over a given time) to implement the data management and reallocation regime from *Christenson* to reduce bottlenecks, identify where and how *Wolf* should relocate data, and improve the speed and performance of *Wolf*’s drives. Ex. C ¶ 223.

A POSITA would have had a reasonable expectation of success in implementing *Christenson*’s HOT and COLD upper and lower thresholds, as doing so involves nothing more than elementary mathematics. Ex. C ¶ 224. Namely, computing these thresholds would simply involve (1) computing an average data access frequency for data on disk (*e.g.*, Ex. PA-15, 8:12-33), (2) identifying the upper and lower thresholds to use as a matter of choice (*see id.*), (3) adding

and subtracting these thresholds to the average (*id.*, 8:34-9:20), and (4) comparing a given datum's access frequency to the resulting number (*id.*, 8:34-9:20, Fig. 3B). Ex. C ¶ 224. Implementing *Christenson*'s HOT and COLD, communication-based thresholds would thus have amounted to nothing more than applying known elements (*Christenson*'s thresholds) to existing platforms (*Wolf*) to achieve predictable results. Ex. C ¶ 224.

4. Claim 10

Claim [10] depends from claim 1 and recites “*wherein reallocating at least some data on said storage device is based on usage patterns of said data.*” Ex. A, claim 10. *Wolf* and *Christenson* teach this additional limitation of claim 10. Ex. C ¶ 225.

As discussed above for claims 5 and 8, *Christenson* reallocates data on a storage device (“*reallocating at least some data on said storage device*”) based on its HOT and COLD designations. *See, e.g.*, [5], [8], *supra*; *see also* Ex. PA-15, 2:51-62, 3:9-45, 11:1-47, Fig. 6. Specifically, *Christenson* “move[s]” (“*reallocates*”) “all the HOT data ... to the HOT partition” on disk and “move[s]” the “identified COLD data ... to the COLD partition.” *Id.*, 9:26-54. When a storage solution uses several disks, *Christenson* further designates the disks themselves as HOT (overutilized), COLD (underutilized), or neither and then moves data to HOT partitions on colder drives. *See, e.g., id.*, 2:63-3:8, 3:21-45, 10:15-53, 11:1-16, Fig. 6. This “balance[s]” the load on the system and reduces bottlenecks. *Id.*, 8:34-9:20, 11:1-16. *Christenson* accomplishes this by defining a “utilization factor” for each disk as a function of the frequency of access for data on the disk and ensuring that the HOT partitions are sufficiently large. *See, e.g., id.*, 3:21-45, 10:14-53; Ex. C ¶ 226.

A POSITA would have understood *Christenson*'s HOT and COLD designations reflect “*usage patterns of said data.*” Ex. C ¶ 227. For example, HOT data reflects data whose “frequency of access is greater than the average by an identification threshold.” Ex. PA-15, 8:34-

9:20. Conversely, COLD data represents data whose “frequency of access for the particular data portion is less than the average by an identification threshold amount.” *Id.* Data falling between these upper and lower limits is designated “neither” HOLD nor COLD (e.g., neutral) and is considered “average.” *Id.* Because these classifications depend on frequency measures reflecting patterns in how users access data—both for individual data portions and all data on disk as a whole—a POSITA would have understood *Christenson*’s reallocation process “*reall[oc]at[es] at least some data on said storage device is based on usage patterns of said data.*” Ex. C ¶ 227. Further, *Christenson*’s designations of HOT and COLD disks also reflect “*usage patterns of said data,*” as these designations depend on how often data is accessed on a particular disk in the aggregate. Ex. C ¶ 227.

A POSITA would have been motivated to implement *Christenson*’s reallocation scheme within *Wolf* for the reasons discussed above. *See, e.g.,* [5], [8], *supra*; Ex. C ¶ 228. For example, a POSITA would have been motivated to reallocate *Wolf*’s data based on *Christenson*’s HOT and COLD designations to improve the speed, efficiency, and performance of *Wolf*’s disk drives. Ex. C ¶ 228. *Wolf* already contemplates relocating data to “safe” locations based on slow access times (*see, e.g.,* [1c], [2]-[3], *supra*), and *Christenson* addresses this problem by ensuring that frequently accessed (HOT) data would be moved to a suitable location to reduce these access times. Ex. C ¶ 228. As *Christenson* instructs, a POSITA would have appreciated how bottlenecks for HOT data impact drive performance. Ex. PA-15, 1:50-2:22. Thus, a POSITA in possession of *Wolf* would have been motivated to seek out references, like *Christenson*, teaching ways of reducing access times for slow data transfers. Ex. C ¶ 228. *Christenson*’s solution via HOT and COLD data designations, data partitions, and drive classifications solve this problem, and a

POSITA would have been motivated to implement this scheme in *Wolf* to further improve its drive performance. Ex. C ¶ 228.

A POSITA would have had a reasonable expectation of success in implementing *Christenson*'s usage-based reallocation scheme in *Wolf* for the reasons discussed above. *See, e.g.,* [5], [8], *supra*; Ex. C ¶ 229. For example, *Wolf* already discloses reallocating data based on various performance factors such as slow access times and detected defects. *See, e.g.,* [1c], [2]-[3], *supra*. Thus, a POSITA would have understood how to implement *Christenson*'s frequency-based reallocation methods within *Wolf*'s framework, as doing so would simply involve computing simple access frequency calculations (*see* [5], [8]), establishing HOT and COLD partitions on a drive (*see* Ex. PA-15, 3:9-20, 9:26-54), and undertaking the same reallocation process *Wolf* already envisions. Ex. C ¶ 229. Thus, implementing *Christenson*'s reallocation scheme in *Wolf* would have amounted to nothing more than applying known calculations and partitioning techniques (as in *Christenson*) to *Wolf*'s existing data reallocation solution to achieve predictable results. Ex. C ¶ 229.

5. Claim 26

Claim [26] depends from claim 25 and recites “*wherein intercepting said communications comprises determining access location, access frequency, and access duration for said data on said storage device.*” Ex. A, claim 26. *Wolf* and *Christenson* teach this additional limitation of claim 26. Ex. C ¶ 230. For example, *Wolf* and *Christenson* each teach “*determining access location*” for the reasons discussed in claims 1, 5, and 6. *See* [1pre]-[1c], [5], [6], *supra*. Further, *Christenson* determines an “*access frequency*” for the reasons discussed in claim 5. *See* [5], *supra*. And *Wolf* teaches determining an “*access duration*” for the reasons discussed for claim 6. *See* [6], *supra*.

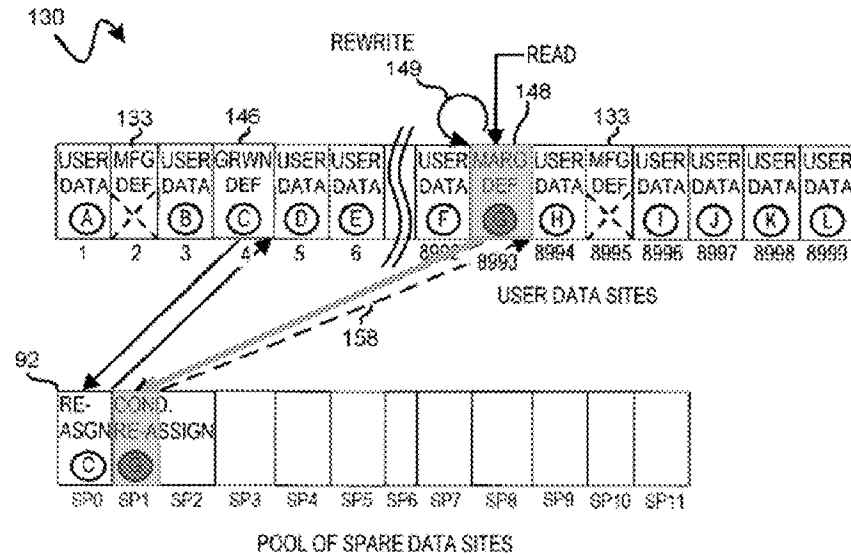
H. Ground 8: Claim 9 Would Have Been Obvious Over *Wolf* and *Schibilla*

1. Claim 9

Claim [9] depends from claim 1 and recites “*wherein responding to said declining performance of said storage device further comprises automatically backing-up data stored on said storage device.*” Ex. A, claim 9. *Wolf* and *Schibilla* teach this additional limitation of claim 9. Ex. C ¶ 231.

As discussed above in section VII.A.8, for example, *Schibilla* discloses “*responding to said declining performance of said storage device further comprises automatically backing-up data stored on said storage device.*” See § VII.A.8, *supra*. For example, *Schibilla* explains how its automatic conditional reassignment creates a “backup copy of the data” on the disk (“*automatically backing-up data stored on said storage device*”). Ex. PA-1, 6:62-7:10, 8:30-49; *see also* [1c], *supra*; Ex. C ¶ 232. As discussed above, this conditional reassignment occurs when a disk’s performance breaches a threshold (“*responding to said declining performance of said storage device*”). See § VII.A.4.c-d (Ground 1 [1b]-[1c]), *supra*; Ex. C ¶ 232.

By copying the impacted data to a separate portion on the drive, *Schibilla*’s conditional reassignment “cop[ies] the data block to [a] spare data site,” which “provid[es] a data backup should a failure occur during the rewrite operation to the marginally defective data site.” Ex. PA-1, 8:30-49; *see also, e.g., id.*, 6:62-7:10, 10:15-41; Ex. C ¶ 233; § VII.A.4.d, *supra*. Before the conditional reassignment completes, data in marginally defective sites exist in both the original site and the pool of spare data sites, such as data block G being recorded in data sites 8993 and SP1 below:



Ex. PA-1, Fig. 5; Ex. C ¶ 233. Because this conditional reassignment “cop[ies]” data to a spare site to “provid[e] a data backup” of the data block, a POSITA would have understood *Schibilla* discloses “*automatically backing-up data stored on said storage device.*” Ex. C ¶ 233.

To the extent Patent Owner disputes *Schibilla*’s disclosures in this regard, it would have been obvious to a POSITA to automatically “*backup*” data in the face of declining performance. Ex. C ¶ 234. For example, a POSITA would have found it obvious to automatically backup at least a portion of data in *Wolf* and/or *Schibilla*’s disk drive (e.g., data stored in marginally defective sites and other contiguous data) in response to detecting that the drive’s performance crossed some performance threshold, as *Wolf* and *Schibilla* disclose. Ex. C ¶ 234. At the time of the ’832 patent, it was common and well-known to “*automatically back[]-up data*” in response to declining performance in disk drives. *See* Ex. C ¶ 234; *see also*, e.g., PA-12 ¶ [0031]. A POSITA would therefore have found it obvious, in light of *Wolf*, *Schibilla*, and this background knowledge, to back up data in response to detecting a performance decline (e.g., a potential device failure) to storage to preserve data and avoid data loss. Ex. C ¶ 234.

A POSITA would have been motivated to automatically backup *Wolf*'s data, as taught in *Schibilla*, in response to performance drops for several reasons. Ex. C ¶ 235. For example, both *Schibilla* and *Wolf* disclose monitoring a disk's performance and reallocating data based on "defective or marginally defective" regions on disk. *See, e.g.*, Ex. PA-1, Abstract, 1:20-35; Ex. PA-14, Abstract, 2:62-3:19, 7:12-37. A POSITA in possession of *Wolf* would thus have been motivated to seek out references, like *Schibilla*, explaining how to conduct this process effectively. Ex. C ¶ 235. For example, *Schibilla* teaches how its conditional reallocation and backup scheme addresses problems occurring when power issues or other interruptions arise during data reassignment. *See, e.g.*, Ex. PA-1, 3:5-13, 6:62-7:22. A POSITA would thus have been motivated to implement routines, such as *Schibilla*'s conditional backup scheme or copying data to spare drives more generally, to help prevent data loss due to power interruptions or failing drives. Ex. C ¶ 235. Indeed, *Schibilla* itself teaches this motivation. Ex. C ¶ 235.

Further, a POSITA would have understood that poor disk drive performance might signal a failing drive (as taught in *Wolf* and *Schibilla*) and thus would have been motivated (and found it obvious) to copy data from a failing drive into a more secure location. Ex. C ¶ 236. Indeed, many references at the time of the '832 already disclosed backing up data in response to a predicted drive failure. *See* Ex. C ¶ 236; Exs. PA-12; PA-13. Thus, a POSITA would have been motivated to apply these techniques to *Wolf* and *Schibilla* to help improve the security of *Wolf*'s reallocation scheme and fulfill *Schibilla*'s stated goal of preventing data loss in the case of an expected drive malfunction. Ex. C ¶ 236.

It would have been within the skill level of a POSITA to backup data in response to performance issues. Ex. C ¶ 237. As discussed, *Wolf* and *Schibilla* already teach monitoring disk drives, detecting waning performance, and reallocating and backing up data in response. *See,*

e.g., [1pre]-[1c] (*Wolf*); § VII.A.8.b-d, *supra* (*Schibilla*). And as explained, backing up data was a prevalent feature of the art at the time of the '832 patent. Ex. C ¶ 237; Ex. PA-12 ¶¶ 0030-0031; Ex. PA-13, 4:13-38. Backing up data in alternative locations in response to declining disk performance would have thus reflected nothing more than the application of known solutions (*e.g.*, backing up data, monitoring disk performance) to existing methods (*e.g.*, *Wolf*'s disk performance monitoring and data management methods) to yield predictable results (*e.g.*, backed up data in response to performance declines). Ex. C ¶ 237.

I. Ground 9: Claim 11 Would Have Been Obvious over *Wolf* and *Jensen*

1. Claim 11

Claim [11] depends from claim 1 and recites “*wherein responding to said declining performance of said storage device further comprises defragmenting at least a portion of said storage device.*” Ex. A, claim 11. *Wolf* and *Jensen* teach this additional limitation. Ex. C ¶ 238.

For example, *Wolf* teaches “*responding to said declining performance of said storage device*” for the reasons discussed in Ground 6, [1c]. *See* Ground 6, [1c], *supra*; Ex. C ¶ 239. As part of this process, *Wolf* explains a relocation process that “relocat[es] data to a usable area. PA-14, Abstract; *see also id.* at 7:12-25; Ex. C ¶ 239. While *Wolf* discloses moving the data to a “good area” (*see supra*), it does not expressly disclose “*defragmenting*” the drive after its conditional reassignment process.

Jensen discloses these details. Ex. C ¶ 240. Indeed, *Jensen* describes techniques for defragmenting disk drives in a computer system, particularly those running Windows NT. *See* Ex. PA-5, Abstract, 1:6-9, 2:32-48, 3:49-61. *Jensen*'s defragmentation process generally comprises identifying fragmented files, determining whether the disk contains sufficient free space, moving some of or the entire file to contiguous locations, and consolidating free space. *See* Ex. PA-5, 3:49-4:4, Figs. 4-9, Ex. C ¶ 240; *see also, e.g.*, Ex. PA-5, 8:24-9:30 (describing a

partial and total defragmentation), 9:33-10:8 (describing moving files), 13:65-14:47. This process minimizes file fragmentation and the number of accesses required to transfer files. Ex. PA-5, 3:5-11; Ex. C ¶ 240. Because *Jensen* defragments some or all file content on a disk, a POSITA would have understood *Jensen* to disclose “*defragmenting at least a portion of said storage device.*” Ex. C ¶ 240.

A POSITA would have found it obvious to defragment a portion of a disk drive (as in *Jensen*) in response to detecting that a disk’s performance has declined (as in *Wolf*). Ex. C ¶ 241. As *Jensen* teaches, certain operating systems, including Windows NT, commonly fragmented data. *See, e.g.*, Ex. PA-5, 1:41-2:28, 2:40-48, 3:49-61; Ex. C ¶ 241. While *Wolf* discloses using a DOS disk (PA-14, 4:52-59), a POSITA would have understood that operating systems like Windows NT and DOS would have encountered the same performance issues. Ex. C ¶ 241. This would have led a POSITA in possession of *Wolf* to *Jensen*, whose disclosure addresses these issues by efficiently defragmenting drives for certain operating systems. Ex. C ¶ 241; Ex. PA-5, 2:40-48, 3:49-61, 5:22-44, 5:66-6:12. It was thus well understood at the time of the ’832 patent to defragment disk drives to keep data contiguous and increase performance. *See* Ex. C ¶ 241; Ex. PA-5, 2:40-48.

A POSITA would have had a reasonable expectation of success in defragmenting *Wolf*’s disk drive in response to a detected decline in performance. Ex. C ¶ 242. As discussed, *Wolf* already monitors disk performance and responds by potentially reallocating data on a disk. *See, e.g.* Ground 6, [1c], *supra*; Ex. C ¶ 242. Disk defragmentation was a ubiquitous process at the time of the ’832 patent’s earliest priority date. Ex. C ¶ 242. Thus, defragmenting some or all of *Wolf*’s disk drive would reflect nothing more than applying known processes (e.g., disk

defragmentation) to existing methods (e.g., *Wolf*'s disk monitoring and reallocation processes) to achieve predictable results. Ex. C ¶ 242.

J. Ground 10: In the Alternative, Claims 6 and 26 Would Have Been Obvious over *Schibilla, Nagar, and Georgiou*

Claim 6 would have been obvious over *Schibilla, Nagar, and Hospodor* for the reasons discussed above. *See, e.g., VII.B.4, supra*. Nevertheless, in the event Patent Owner disputes this combination, claims 6 and 26 would alternatively have been obvious over *Schibilla, Nagar, and Georgiou*. Ex. C ¶¶ 245-250.

1. Overview of *Georgiou*

Georgiou is directed to optimizing a disk drive's power consumption. Ex. PA-16, Abstract. The system monitors a disk's durations of activity, predicting when the disk will be active or inactive in the future, and adjusting the drive's rotational speed to better conserve power. *Id.* As part of this process, *Georgiou* determines a "duration" for how long a user has accessed data on the disk as well as the "duration" for each of the disk driver's operational states. *See, e.g., Ex. PA-16, 2:56-65; Ex. C ¶ 244.*

2. Claim 6

Claim [6] depends from claim 1 and recites "*determining an access location on said storage device and an access duration for data stored thereon, based on said intercepted communications.*" Ex. A, claim 6. *Schibilla-Nagar* and *Georgiou* teach this additional limitation of claim 6. Ex. C ¶ 245.

Schibilla-Nagar discloses "*determining an access location on said storage device*" for the same reasons discussed above for claims 1 and 5. *See, e.g., §§ VII.A.4, VII.A.6, supra; Ex. C ¶ 246.* For example, *Schibilla* discloses determining an "*access location*" for read or write operations by identifying particular "site[s]" those operations reference. *See § VII.A.4, supra;*

see also, e.g., Ex. PA-1, Abstract, 6:36-7:10, 8:5-16, Figs 3-5. This enables *Schibilla* to monitor the performance of a particular “site” or “address” in data. *See, e.g.*, Ex. PA-1, 9:65-10:55. *Schibilla* thus monitors operations performed on a site-by-site basis. *Id.*, Figs. 1, 5. A POSITA would have understood that *Schibilla*’s monitoring of particular sites on disk reflects the claimed “access location on said storage device.” Ex. C ¶ 246.

Georgiou teaches determining “an access duration for data stored thereon.” Ex. C ¶ 247. For example, *Georgiou* describes ways for optimizing power consumption used by a disk drive. *See* PA-16, Abstract. *Georgiou* uses control logic to determine, access times, the “duration of current disk activity,” and the “duration” of a disk driver’s operational state. *Id.*, 2:56-3:6. *Georgiou* uses logs histories of disk activity and inactivity to predict future periods of activity and inactivity for the drive. *See, e.g., id.*, 2:36-46. Using these predictive algorithms, *Georgiou* alters the drive’s rotational speed to improve a disk’s power consumption profile. *Id.* Because *Georgiou* discloses measuring the “duration” disk activity and the “duration” of a disk driver’s operational state over a period of time, a POSITA would have understood *Georgiou* teaches “determining ... an access duration for data stored thereon.” Ex. C ¶ 247.

A POSITA would have been motivated to monitor the access duration of data to reduce the power consumption of *Schibilla*’s drive and expand the number of performance metrics (e.g., access duration) tracked by *Schibilla*’s system. Ex. C ¶ 248. As *Georgiou* instructs, disk drives consume considerable power, a consideration particularly acute in laptop settings. *See, e.g.*, Ex. PA-16, 1:11-34. A POSITA would have been aware of this issue and thus would have sought out references, such as *Georgiou*, to discover ways of reducing the power consumption of *Schibilla*’s disk drive. *Schibilla* and *Georgiou* both describe ways of improving disk performance. Ex. C ¶ 248. A POSITA in possession of *Schibilla-Nagar* would thus have sought

out additional ways of improving disk drive performance, like *Georgiou*, to account for other methods of improving a real disk drive. Ex. C ¶ 248.

A POSITA would have had a reasonable expectation of success in determining access duration, as doing so would involve nothing more than determining how long a user interacted with a particular file or files on disk. Ex. C ¶ 249. Given that *Schibilla* already monitors read and write operations passing to disk, and that *Nagar* teaches measuring access frequencies for files or data on disk (*see, e.g.,* § VII.A.6, *supra*), measuring the duration of time a user interacted with a file or data on disk would have been a straightforward task. Ex. C ¶ 249. Measuring access duration in *Schibilla-Nagar* would thus have involved applying known techniques (*Georgiou*'s access duration determination) with known technology (*Schibilla-Nagar*'s disk monitoring system) to achieve predictable results. Ex. C ¶ 249.

3. Claim 26

Claim [26] depends from claim 25 and recites “*wherein intercepting said communications comprises determining access location, access frequency, and access duration for said data on said storage device.*” Ex. A, claim 26. *Schibilla-Nagar* in view of *Georgiou* teaches this additional limitation of claim 26. Ex. C ¶ 250. For example, *Schibilla-Nagar* teaches determining an “*access location*” and an “*access frequency*” for the reasons discussed for claim 5. *See* § VII.A.6, *supra*. And *Georgiou* teaches determining an “*access duration*” for the reasons discussed for claim 6. *See* Ground 10, [6] (§ VII.J.2), *supra*.

K. Ground 11: In the Alternative, Claim 6 Would Have Been Obvious over *Wolf* and *Georgiou*

Claim 6 would have been obvious over *Wolf* for the reasons discussed above. *See, e.g.,* VII.F.5, *supra*. Nevertheless, in the event Patent Owner disputes this combination, claim 6 would alternatively have been obvious over *Wolf* and *Georgiou*. Ex. C ¶ 251.

1. Claim 6

Claim [6] depends from claim 1 and recites “*determining an access location on said storage device and an access duration for data stored thereon, based on said intercepted communications.*” Ex. A, claim 6. *Wolf* and *Georgiou* teach this additional limitation of claim 6. Ex. C ¶ 252.

For example, *Wolf* discloses “*determining an access location on said storage device*” for the same reasons discussed above for claims 1 and 5. *See, e.g.,* §§ VII.F.6, *supra*; Ex. C ¶ 253. *Wolf* monitors data transfer times and errors occurring in “defective or marginally defective storage areas,” marks these areas as defective, and “relocat[es] the data to a usable area.” Ex. PA-14, Abstract. *Wolf* records this information in a log to create “a history of the [storage] medium per unit area, e.g., per sector” (e.g., a “*location*”). *Id.*, 3:40-53. This log reflects the historical access performance of a particular sector, area, or location on disk. *See id.*, 7:12-25 (maintaining and analyzing a “history of such events sorted by disk unit areas, such as sectors”), 12:20-45 (explaining how the disk-read timer inspects specified “memory addresses” for changes when counting time), 12:46-68 (same), 13:1-8, 13:49-52 (saving the “address” of an instruction), 14:46-56, 15:51-65, 17:15-38 (discussing how the analysis program attempts to “correct bad locations” on disk based errors detected at specification locations), 20:24-41 (logging read times for sectors and blocks of sectors). Ex. C ¶ 253. *Wolf* uses this information for various purposes, such as maintaining histories of “the location at which [an] error occurred.” Ex. PA-14, 17:23-55. *Wolf* recites this disclosure across several claims. *See id.*, 22:11-52 (reciting “logging information concerning each operation for retrieving data during which an error condition occurred, the information including the location on the medium from which data was retrieved”), 23:5-47, 24:28-39, 24:51-25:24, 25:57-26:31, 26:53-27:26, 28:16-58; Ex. C ¶ 253.

Georgiou teaches determining “an access duration for data stored thereon.” Ex. C ¶ 254.

As discussed above, for example, *Georgiou* describes ways for optimizing power consumption used by a disk drive. *See* PA-16, Abstract. *Georgiou* uses control logic to determine, access times, the “duration of current disk activity,” and the “duration” of a disk driver’s operational state. *Id.*, 2:56-3:6. *Georgiou* uses logs histories of disk activity and inactivity to predict future periods of activity and inactivity for the drive. *See, e.g., id.*, 2:36-46. Using these predictive algorithms, *Georgiou* alters the drive’s rotational speed to improve a disk’s power consumption profile. *Id.* Because *Georgiou* discloses measuring the “duration” of disk activity and the “duration” of a disk driver’s operational state over a period of time, a POSITA would have understood *Georgiou* teaches “determining ... an access duration for data stored thereon.” Ex. C ¶ 254.

A POSITA would have been motivated to monitor the access duration of data to reduce the power consumption of *Wolf*’s drive and expand the number of performance metrics (e.g., access duration) tracked by *Wolf*’s system. Ex. C ¶ 255. As *Georgiou* instructs, disk drives consume considerable power, a consideration particularly acute in laptop settings. *See, e.g.,* Ex. PA-16, 1:11-34. A POSITA would have been aware of this issue and thus would have sought out references, such as *Georgiou*, to discover ways of reducing the power consumption of *Wolf*’s disk drive. *Wolf* and *Georgiou* both describe ways of improving disk performance, and both monitor similar performance metrics such as access times. *See, e.g.,* Ex. PA-14, 3:22-39, 6:57-7:11, 7:55-58, 9:17-55, 20:24-41; Ex. PA-16, 2:36-46; Ex. C ¶ 255. A POSITA in possession of *Wolf* would thus have sought out additional ways of improving disk drive performance, like *Georgiou*, to account for other methods of improving a real disk drive. Ex. C ¶ 255.

A POSITA would have had a reasonable expectation of success in determining access duration, as doing so would involve nothing more than determining how long a user interacted with a particular file or files on disk. Ex. C ¶ 256. Given that *Wolf* already monitors access times (*see, e.g.*, [1pre]-[1c], [2], *supra*), measuring the duration of time a user interacted with a file or data on disk would have been a straightforward task. Ex. C ¶ 256. Measuring access duration in *Wolf* would thus have involved applying known techniques (*Georgiou*'s access duration determination) with known technology (*Wolf*'s disk monitoring system) to achieve predictable results. Ex. C ¶ 256.

L. Ground 12: In the Alternative, Claim 26 Would Have Been Obvious over *Wolf*, *Christenson*, and *Georgiou*

Claim 26 would have been obvious over *Wolf* and *Christenson* for the reasons discussed above. *See, e.g.*, VII.G.5, *supra*. Nevertheless, in the event Patent Owner disputes this combination, claim 26 would alternatively have been obvious over *Wolf*, *Christenson*, and *Georgiou*. Ex. C ¶ 257.

1. Claim 26

Claim [26] depends from claim 25 and recites “*wherein intercepting said communications comprises determining access location, access frequency, and access duration for said data on said storage device.*” Ex. A, claim 26. *Wolf* and *Christenson* in view of *Georgiou* teaches this additional limitation of claim 26. Ex. C ¶ 258. For example, *Wolf* and *Christenson* teach determining an “*access location*” and an “*access frequency*” for the reasons discussed for claims 1 and 5. *See* §§ VII.F.2, VII.G.2, *supra*. And *Georgiou* teaches determining an “*access duration*” for the reasons discussed for claim 6. *See* [6] (§ VII.K.1), *supra*.

It would have been obvious to combine *Wolf* and *Christenson* with *Georgiou* for the same reasons discussed in those sections. *See, e.g.*, §§ V.F, V.K; Ex. C ¶ 259. For example, a POSITA

would have been motivated to implement *Georgiou*'s power-consumption reduction routines into *Wolf* and *Christenson* to improve the power efficiency of *Wolf*'s drive and expand the performance metrics *Wolf* and *Christenson* assess when monitoring disk performance to better understand the disk's status. Ex. C ¶ 259. A POSITA would have had a reasonable expectation of success in making this combination to measure access duration, as doing so would simply involve combining known computations (access duration) to existing solutions (*Wolf*'s monitoring system and *Christenson*'s reallocation scheme) to achieve predictable results. Ex. C ¶ 259.

VIII. CONCLUSION

For the foregoing reasons, NetApp requests reexamination and cancellation of claims 1-26 of the '832 patent.

NetApp is willing to provide appropriate assistance to permit the Examiner to address and decide the issues presented by this Request. As the M.P.E.P. explains, the Examiner may, when appropriate, cut and paste claim charts or other material within the Request to incorporate them within the body of an Office Action. M.P.E.P. § 2262. NetApp is, thus, through the undersigned counsel, available to provide the Examiner with a digital copy of this Request, or any portion of it, in response to a request by email or phone. NetApp also understands that the Examiner may, in appropriate circumstances, set forth specific rejections in an Office Action and incorporate by reference NetApp's reasons for the proposed rejections, if the Examiner agrees with the proposed rejections and reasons supporting them.

If there is any additional fee due in connection with this filing that is not already submitted herewith, please charge the fee to Deposit Account No. 06-0916.

Respectfully submitted,
FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, LLP

Dated: October 15, 2025

By: /Cory C. Bell/
Cory C. Bell
Reg. No. 75,096

**Attachments: (1) Certificate of Service to Patent Owner
(2) Information Disclosure Statement Form PTO/SB/08
(3) Exhibits listed on page iii of this Request**