

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

AMERICAN AIRLINES, INC. AND SOUTHWEST AIRLINES CO.,

Petitioners

v.

INTELLECTUAL VENTURES II LLC,

Patent Owner.

IPR2026-00043
U.S. Patent No. 7,721,282

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 7,721,282**

TABLE OF CONTENTS

TABLE OF AUTHORITIES	iv
EXHIBIT LIST	v
I. MANDATORY NOTICES UNDER 37 C.F.R. § 42.8.....	1
II. GROUNDS FOR STANDING.....	4
III. IDENTIFICATION OF CHALLENGES AND RELIEF REQUESTED	5
IV. THE '282 PATENT	7
A. Overview	7
B. Effective Filing Date	10
C. Level of Ordinary Skill	11
V. CLAIM CONSTRUCTION	11
VI. OVERVIEW OF THE ART	12
A. Birse (Ex.1005)	15
B. Ballew (Ex.1006).....	20
C. Wu (Ex.1007).....	23
D. Kleinschnitz (Ex.1008).....	25
VII. GROUND 1: CLAIMS 1, 3-10, 13-16, 19-26, 29, 30 ARE OBVIOUS OVER BIRSE	26
A. Birse meets all claim elements.	26
1. Claims 1 and 15.....	26
2. Claim 16.....	56
3. Claims 3 and 19.....	58
4. Claims 4 and 20.....	59

5.	Claims 5 and 21.....	59
6.	Claims 6 and 22.....	61
7.	Claims 7 and 23.....	62
8.	Claims 8 and 24.....	64
9.	Claims 9 and 25.....	65
10.	Claim 10.....	66
11.	Claim 26.....	67
12.	Claims 13 and 29.....	67
13.	Claims 14 and 30.....	68
VIII.	GROUND 2: CLAIMS 2, 11, 12, 18, 27, AND 28 ARE OBVIOUS OVER THE COMBINATION OF BIRSE AND BALLEW	74
A.	Motivation to Combine	74
B.	Birse and Ballew meet all claim elements.	75
1.	Claims 2 and 18.....	75
2.	Claims 11 and 27.....	76
3.	Claims 12 and 28.....	79
IX.	GROUND 3: CLAIMS 8 AND 24 ARE OBVIOUS OVER THE COMBINATION OF BIRSE AND WU.....	80
A.	Motivation to Combine	80
B.	Birse and Wu meet all claim elements.	82
1.	Claims 8 and 24.....	82
X.	GROUND 4: CLAIMS 9, 17, AND 25 ARE OBVIOUS OVER THE COMBINATION OF BIRSE AND KLEINSCHNITZ.....	83
A.	Motivation to Combine	83

Petition for *Inter Partes* Review of
U.S. Patent No. 7,721,282

B.	Birse and Kleinschnitz meet all claim elements.	85
1.	Claims 9 and 25.....	85
2.	Claim 17.....	86
XI.	CONCLUSION.....	87

TABLE OF AUTHORITIES

Cases

<i>Mylan Pharmaceuticals Inc., v. Regeneron Pharmaceuticals, Inc.</i> , IPR2022-01226, Paper 22 (PTAB Jan. 11, 2023).....	6
<i>Phillips v. AWH Corp.</i> , 415 F.3d 1303 (Fed. Cir. 2005).....	11
<i>Studiengesellschaft Kohle v. Shell Oil Co.</i> , 112 F.3d 1561 (Fed. Cir. 1997).....	11
<i>Toyota Motor Corp. v. Cellport Systems, Inc.</i> , IPR2015-00633, Paper 11 (Aug. 14, 2015)	12
<i>Wellman, Inc. v. Eastman Chemical Co.</i> , 642 F.3d 1355 (Fed. Cir. 2011).....	12

Rules

37 C.F.R. § 42.104	4
37 C.F.R. §42.100	11

EXHIBIT LIST

Exhibit No.	Description of Document
1001	U.S. Patent No. 7,721,282
1002	Prosecution File History of U.S. Patent No. 7,721,282 (Application No. 11/395,816)
1003	Expert Declaration of Erez Zadok, Ph.D.
1004	Erez Zadok, Ph.D. Curriculum Vitae
1005	U.S. Patent No. 7,089,300, entitled “Method and Apparatus for Administering the Operating System of a Net-Booted Environment,” filed on October 18, 1999, and issued on August 8, 2006 to Cameron Stuart Birse, et al. (“Birse”)
1006	U.S. Patent Application Publication No. 2005/0235092, entitled “High Performance Computing System and Method,” filed on April 15, 2004, and published on October 20, 2005 to James D. Ballew, et al. (“Ballew”)
1007	U.S. Patent No. 6,981,114, entitled “Snapshot Reconstruction From An Existing Snapshot and One or More Modification Logs,” filed on October 16, 2002, and issued on December 27, 2005 to Weibao Wu, et al. (“Wu”)
1008	U.S. Patent Application Publication No. 2003/0191911, entitled “Using Disassociated Images for Computer and Storage Resource Management,” filed on April 2, 2003, and published on October 9, 2003 to Donald Kleinschnitz, Jr., et al. (“Kleinschnitz”)
1009	File History for U.S. Patent Application No. 11/026,622 (the “’622 Application”, now abandoned)
1010	Claim Listing
1011	Microsoft Computer Dictionary, 3rd ed., 1997, excerpts (“Microsoft Computer Dictionary”)

Exhibit No.	Description of Document
1012	Internet Small Computer Systems Interface (iSCSI), RFC 3720, IETF, April 2004, https://www.rfc-editor.org/rfc/pdf/rfc3720.txt.pdf
1013	File System Design for an NFS File Server Appliance, Dave Hitz et al., Network Appliance, USENIX 1994 (“Hitz”)
1014	SnapMirror: File System Based Asynchronous Mirroring for Disaster Recovery, Hugo Patterson et al., Network Appliance, USENIX 2002 (“Patterson I”)
1015	U.S. Patent 6,668,264B1 to Patterson et al., Filed Apr. 3, 2001, Issued Dec. 23, 2003, titled “Resynchronization of a target volume with a source volume” (“Patterson II”)
1016	U.S. Patent 7,334,095B1 to Fair et al. Filed Apr. 30, 2004 (“Fair”)
1017	U.S. Patent 8,209,680B1 to Le et al., Filed June 30, 2003, Issued June 26, 2012, titled “System and method for disk imaging on diverse computers” (“Le”)
1018	Complaint, <i>Intellectual Ventures I LLC et al. v. Southwest Airlines Co.</i> , No. 7:24-cv-00277 (W.D. Tex. Nov. 2, 2024) [Dkt. 1]
1019	Complaint, <i>Intellectual Ventures I LLC et al. v. American Airlines, Inc.</i> , No. 4:24-cv-980-ALM (E.D. Tex. Nov. 2, 2024) [Dkt. 1]
1020	Second Amended Complaint (redacted), <i>Intellectual Ventures I LLC et al. v. Southwest Airlines Co.</i> , No. 7:24-cv-277 (W.D. Tex. Sept. 10, 2025) (Dkt. 67)
1021	Amended Complaint, <i>Intellectual Ventures I LLC et al. v. American Airlines, Inc.</i> , No. 4:24-cv-980 (E.D. Tex. Aug. 25, 2025) [Dkt. 76]
1022	Complaint, <i>Intellectual Ventures I LLC et al. v. American Airlines, Inc.</i> , No. 4:25-cv-00576-ALM (E.D. Tex. May 30, 2025)

Petitioners American Airlines, Inc. (“American”) and Southwest Airlines Co. (“Southwest”) (collectively “Petitioners”) petition for *inter partes* review (“IPR”) of claims 1-30 of U.S. Patent No. 7,721,282 (the “’282 Patent”) owned by Intellectual Ventures II LLC (“Patent Owner”).

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

A. Real Parties-in-Interest Under § 42.8(b)(1)

American is a real party in interest to this petition. American is a wholly owned subsidiary of American Airlines Group, Inc., which is not a real party in interest to this petition. No unnamed entity is funding, controlling, or otherwise has an opportunity to control or direct this petition or American’s participation in any resulting IPR.

Southwest is also a real party in interest to this petition. No unnamed entity is funding, controlling, or otherwise has an opportunity to control or direct this petition or Southwest’s participation in any resulting IPR.

B. Related Matters Under § 42.8(b)(2)

To the best of Petitioners’ knowledge, the ’282 Patent has been involved in the following related matters:

- *Intellectual Ventures I LLC et al. v. Nationwide Mutual Insurance Company*, No. 3:25-cv-00632 (N.D. Tex. filed March 15, 2025)

Petition for *Inter Partes* Review of
U.S. Patent No. 7,721,282

- *Intellectual Ventures I LLC et al. v. The Bank of New York Mellon Corporation*, No. 3:25-cv-00631 (N.D. Tex. filed March 15, 2025)
- *Intellectual Ventures I LLC et al. v. The Home Depot, Inc. et al.*, No. 1:25-cv-01147 (W.D. Tex. filed July 23, 2025)

The '282 Patent is also asserted in the following cases against Petitioners:

- *Intellectual Ventures I LLC et al. v. American Airlines, Inc.*, No. 24-cv-00980 (E.D. Tex. November 2, 2024)
- *Intellectual Ventures I LLC et al. v. American Airlines, Inc.*, No. 4:25-cv-00576 (E.D. Tex. May 30, 2025)
- *Intellectual Ventures I LLC et al. v. Southwest Airlines, Co.*, No. 24-cv-00277 (W.D. Tex. November 2, 2024)

Petitioners are not aware of any other civil actions or proceedings filed in connection with the '282 Patent.

C. Lead and Backup Counsel Under § 42.8(b)(3)

Lead Counsel	Backup Counsel
Keith D. Harden (Reg. No. 74,472) kharden@munckwilson.com Munck Wilson Mandala, LLP 2000 McKinney Ave., Ste. 1900 Dallas, Texas 75201 Tel: (972) 628-3600 Fax: (972) 628-3616	Daniel E. Venglarik (Reg. No. 39,409) dvenglarik@munckwilson.com Munck Wilson Mandala, LLP 2000 McKinney Ave., Ste. 1900 Dallas, Texas 75201 Tel: (972) 628-3600 Fax: (972) 628-3616

Petition for *Inter Partes* Review of
U.S. Patent No. 7,721,282

Lead Counsel	Backup Counsel
	<p>S. Wallace Dunwoody (Texas Bar No. 24040838 - admission <i>pro hac vice</i> to be requested) wdunwoody@munckwilson.com Munck Wilson Mandala, LLP 2000 McKinney Ave., Ste. 1900 Dallas, Texas 75201 Tel: (972) 628-3600 Fax: (972) 628-3616</p> <p>Michael C. Wilson (Texas Bar No. 21704590 - admission <i>pro hac vice</i> to be requested) mwilson@munckwilson.com Munck Wilson Mandala, LLP 2000 McKinney Ave., Ste. 1900 Dallas, Texas 75201 Tel: (972) 628-3600 Fax: (972) 628-3616</p>
	<p>John B. Campbell (Reg. No. 54,665) jcampbell@McKoolSmith.com McKool SMITH, P.C. 303 Colorado Street, Suite 2100 Austin, TX 78701 Tel: (512) 692-8730 Fax: (512) 692-8744</p> <p>Casey Shomaker (Reg. No. 77,998) cshomaker@mckoolsmith.com McKool Smith, P.C. 300 Crescent Court, Suite 1200 Dallas, TX 75201 Tel: (214) 978-4218 Fax: (214) 978-4044</p> <p>Emily Tannenbaum (Reg. No. 80,655)</p>

Lead Counsel	Backup Counsel
	etannenbaum@mckoolsmith.com McKool Smith, P.C. 1301 Avenue of the Americas, Fl. 32 New York, NY 10019 Tel: (212) 402-9441 Fax: (212) 402-9444

An executed power of attorney under 37 C.F.R. § 42.10(b) accompanies this Petition.

D. Service Information Under § 42.8(b)(4)

Petitioners consent to electronic service at the above designated Counsel email addresses and at the following email address:

SWA-IV@munckwilson.com.

Service of any documents via post or hand-delivery may be made at: Munck Wilson Mandala LLP, 2000 McKinney Ave., Ste. 1900, Dallas, TX 75201; Telephone: (972) 628-3600; Facsimile: (972) 628-3616.

II. GROUNDS FOR STANDING

Pursuant to 37 C.F.R. § 42.104(a), Petitioners certify the '282 Patent is available for IPR, and that Petitioners are not barred or estopped from requesting an IPR challenging the claims on the grounds identified in this Petition. This Petition is filed within one year of service of the Complaint. Exs.1018, 1019, 1020, 1021, 1022.

III. IDENTIFICATION OF CHALLENGES AND RELIEF REQUESTED

Petitioners respectfully request review and cancellation of claims 1-30 on the following prior art and statutory grounds:

Prior Art
U.S. 7,089,300 (Birse), issued August 8, 2006, filed October 18, 1999 (Ex.1005); Pre-AIA 35 U.S.C. § 102(e)
U.S. 2005/0235092 (Ballew) published October 20, 2005, filed April 15, 2004 (Ex.1006); Pre-AIA 35 U.S.C. §§ 102(a) and/or (e)
U.S. 6,981,114 (Wu) issued December 27, 2005, filed October 16, 2002 (Ex.1007); Pre-AIA 35 U.S.C. §§ 102(a) and/or (e)
U.S. 2003/0191911 (Kleinschnitz) published October 9, 2003, filed April 2, 2003, and claiming priority to U.S. Provisional No. 60/370,100 filed April 3, 2002 (Ex.1008); Pre-AIA 35 U.S.C. §§ 102(a), (b), and/or (e)

Ground	Challenged Claims	Basis
1	1, 3-10, 13-16, 19-26, 29, 30	§ 103: Birse
2	2, 11, 12, 18, 27, 28	§ 103: Birse and Ballew
3	8, 24	§ 103: Birse and Wu
4	9, 17, 25	§ 103: Birse and Kleinschnitz

Petitioners stipulate that, if trial is instituted, Petitioners will not pursue in the District Court Litigation any ground raised or that could have been reasonably raised in an IPR. *Sotera Wireless, Inc. v. Masimo Corp.*, IPR2020-01019, Paper 12 at 18 (PTAB Dec. 1, 2020).

Petitioners also submit the Declaration of Dr. Erez Zadok (Ex.1003), a qualified expert, in support of the identified grounds. Ex.1003, ¶¶5-28; Ex. 1004.

The identified grounds are not cumulative of the art of record, including the art of record considered by the Examiner. None of the references in the above-grounds were substantively addressed or even cited during prosecution of the '282 Patent, and nothing suggests their teachings relied on herein—much less the combination of those teachings as informed by a POSITA's knowledge and expert testimony—were considered by the Examiner. *Mylan Pharmaceuticals Inc., v. Regeneron Pharmaceuticals, Inc.*, IPR2022-01226, Paper 22 at 27-29 (PTAB Jan. 11, 2023).

U.S. Patent No. 6,751,658 (Haun) (which shares a common specification with Birse) was repeatedly and successfully cited by a different Examiner assigned to the parent application to the '282 Patent (11/026,622 (the "'622 Application'")) to support rejection of all similar claims in the '622 Application, resulting in abandonment. Ex.1009, 94,146,205,263,295. Haun and Birse were not cited during prosecution of the '282 Patent. Further, although Haun was cited in the '622 Application as early as August 24, 2009 (Ex.1009, 92-110), the prosecuting attorneys did not file an Information Disclosure Statement in the '816 Application citing Haun, even though the '816 Application was still pending at the time. *See* Ex.1002 at 87-128. Additionally, while U.S. Patent No. 7,475,274 (Davidson), which shares some overlapping disclosure with Ballew, was cited during prosecution

of the '282 Patent, it was not combined with any of the other references of the identified grounds in this Petition.

IV. THE '282 PATENT

A. Overview

The '282 Patent describes clustered computing involving multiple compute nodes, where nodes are activated by diskless booting protocols or remote software installation to local storage, and where a boot image is required for each node in the cluster. Ex.1001, 1:21-57. Ex.1003, ¶¶132-40. The Background discusses branching-store file systems where a read-only “root” image is accessible by all nodes, and changes made by a node to the root image are stored in a “leaf” image unique to that node. Ex.1001, 1:58-2:5. Ex.1003, ¶¶141-46.

Fig. 1 of the '282 Patent diagrams a system 100 (e.g., “a multi-computer system, such as an HPC [high performance computing] cluster”) having compute nodes 120a-n coupled to a first storage unit 140 and a corresponding second storage unit 150a-n through a corresponding union block device (“UBD”) 130a-n. Ex.1001, 3:62-4:16. To the compute nodes 120a-n, it appears that they have access to their own version of the application environment. However, a separate boot image is not created and stored for each node. Ex.1001, 4:6-10. Ex.1003, ¶¶147-48.

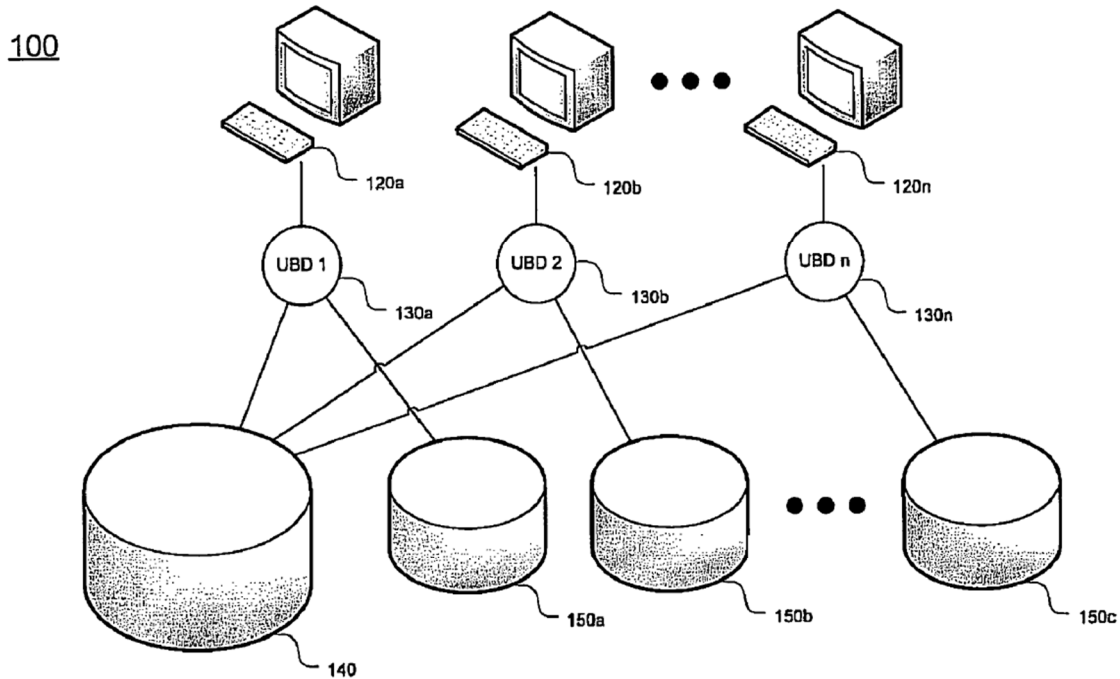


Figure 1

Instead, the first storage unit 140 stores blocks of the root image containing data initially common to the compute nodes 120a-n. Ex.1001, 4:11-16. Each compute node 120a-n also has read-only access to a corresponding second storage unit 150a-n for storing a leaf image, which in the challenged claims includes “new data blocks and changes to the blocks of the root image.” *Id.*, 4:14-18, 7:21-24, 8:12-15. The first storage unit 140 and second storage units 150a-n may each be contained on separate physical storage devices, on separate logical spaces on the same storage device, or any combination thereof. Ex.1001, 4:18-22. Ex.1003, ¶¶149-54.

A compute node 120a-n mounts its instantiation of the application environment via its respective UBD 130a-n, which are “effectively low level drivers that operate as an interface between the first and second storage devices and the file system of each compute node 120a-n.” Ex.1001, 4:41-45. UBDs 130a-n determine what “leaf image” (from the appropriate second storage unit 150) is needed for portions of the image that their respective compute nodes 120a-n have changed, and locate the portions of the image not changed by their respective node (which may reside in the root image or intermediate images comprising versions of the root image, not depicted in Fig. 1). Ex.1001, 4:52-61. Ex.1003, ¶¶157-58.

The blocks of the root image and leaf image are merged to create the node’s image to save space without significantly impacting load times. Ex.1001, 2:49-53, 6:8-16, 6:41-45, 6:56-59. From the node’s perspective, the created image appears as one cohesive image even though a separate and complete image is not actually stored for the node. *Id.*, 6:10-13. Ex.1003, ¶¶138-39, 145, 160.

As shown in FIG. 2, below, the leaf images are also referred to as leaf nodes, which “may be for blocks from configuration files.” Ex. 1001, 5:48-52. Ex.1003, ¶153.

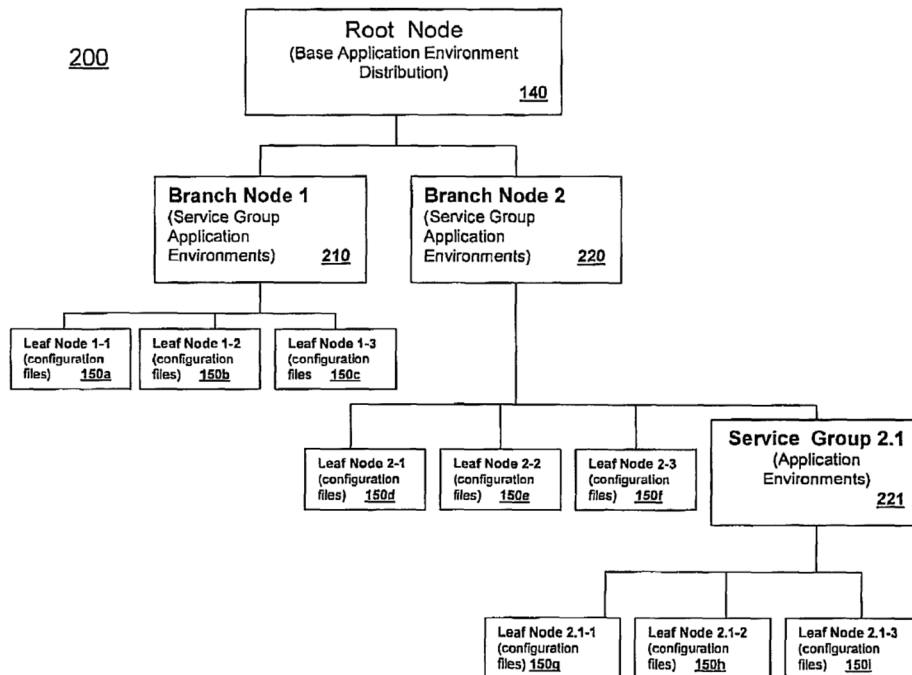


Figure 2

B. Effective Filing Date

The asserted claims of the '282 Patent have a filing date of March 30, 2006. Application No. 11/395,816 (the "'816 Application"), which matured into the '282 Patent, was filed on March 30, 2006 and was a continuation-in-part of the '622 Application (filed December 20, 2004) (abandoned).

However, none of the '282 Patent claims are entitled to the priority date of the '622 Application because the '282 Patent claims subject matter not found in the '622 Application. For example, the '622 Application never mentions "merging the blocks of the root image stored on the first storage unit with the blocks of the leaf image stored on the second storage unit to create the application environment" nor "upon

receiving a write request from the compute node for a sector X, creating an appropriate persistent mapping for sector X” – which are expressly required by all claims of the ’282 Patent. Ex.1001, 5:1-10, 7:11-8:65; *compare* Ex.1001 5:61-6:38, Fig. 3 *with* Ex.1009, 4-40. *See Studiengesellschaft Kohle v. Shell Oil Co.*, 112 F.3d 1561, 1564 (Fed. Cir. 1997). Additionally, claims 1-14 of the ’282 Patent recite use of a “union block device,” which is also not found in the ’622 Application. *See* Ex.1009, 4-40. Therefore, the claims of the ’282 Patent are only entitled to a filing date of March 30, 2006.

C. Level of Ordinary Skill

A POSITA, on or before March 30, 2006, or alternatively on or before December 30, 2004, would have had a bachelor's degree in computer science, computer engineering, or an equivalent degree, and approximately two years of experience working in the fields of networked data storage, file systems, and operating systems. Additional experience might substitute for less education and *vice versa*. Ex.1003, ¶¶34-36.

V. CLAIM CONSTRUCTION

Claim terms are construed according to the *Phillips* standard in an IPR. *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005); 37 C.F.R. §42.100. The analysis in Sections VI-X establishes that the challenged claims are obvious under any reasonable interpretation under the *Phillips* standard, and thus Petitioners submit

that the Board need not resolve any express constructions. *See, e.g., Toyota Motor Corp. v. Cellport Systems, Inc.*, IPR2015-00633, Paper 11 at 16 (Aug. 14, 2015) (citing *Wellman, Inc. v. Eastman Chemical Co.*, 642 F.3d 1355, 1361 (Fed. Cir. 2011)) (claim construction only required where necessary to resolve dispute). Ex.1003, ¶¶213-215.

Should any subsequent discussion of claim interpretation arise here or in the parallel district court proceedings, Petitioners reserve the right to propose constructions and respond to any construction advanced by the Patent Owner, the Board, and/or the Court.

VI. OVERVIEW OF THE ART

Systems and methods for using Internet-connected servers for storing read-only root images and separately storing modified blocks of data (leaf images) and merging modified data with original root image data, were known well before the the '282 Patent. As explained by Dr. Zadok, servers that communicated over the Internet were well known before the mid-2000s. *See* Ex.1003, ¶¶48-85 (citing Microsoft Computer Dictionary, 3rd ed., 1997 (Ex.1011) (describing various Internet networks, communication protocols, and distributed computing environments used by servers) and RFC-3720 (Ex.1012) (describing protocol on top of TCP used to transport data over Internet/IP networks)).

Systems using snapshots with network access and using copy-on-write (CoW) were also known. For example, Hitz (Ex.1013) was a paper on “File System Design for an NFS File Server Appliance” that described a “WAFL (Write Anywhere File Layout)” which was “a file system designed specifically to work in an NFS appliance.” The “primary focus is on the algorithms and data structures that WAFL uses to implement SnapshotsTM, which are read-only clones of the active file system. WAFL uses a copy-on-write technique to minimize the disk space that Snapshots consume.” Ex.1013, 4. Ex.1003, ¶¶86-89. As further explained by Dr. Zadok, Hitz described that snapshots can be created and deleted as needed, offering space saving and improved reliability by using “a copy-on-write technique to avoid duplicating disk blocks.” Ex.1013, 6, 9-11. Ex.1003, ¶¶89-94.

Various other systems used read-only root images with CoW-based systems. *See, e.g.*, Fair (Ex.1016) at 6:18-41, 21:27-31 (describing storing a read-only base snapshot on a source volume and using writable clones), Patterson I (Ex.1014) at 2-3, 8, 12 (describing asynchronous mirrors enabling self-consistent snapshots of data), and Patterson II (Ex.1015) (describing merging/synchronization of a target volume with a source volume where data blocks that are found to not be in the source are copied to a target to replace original source blocks with modified data). Ex.1003, ¶¶95-116.

Another example, Le (Ex.1017), includes a lengthy background section detailing that as of 2003 it was known that a “[a] computer disk can be viewed as a linear list of data blocks called sectors,” that “a sector-based image format describes the state of a disk at the sector (or ‘block’) level,” and that “a file’s content is stored in one or more sectors, called data sectors.” Ex.1017, 1:16-20. Le thus confirms that sectors and blocks relate to the same subject matter. Ex.1003, ¶117. Le also describes that “a sparse image file...is a sector-level representation of a complete disk” and that it “does not store the actual contents of unallocated sectors, since their content is known; it needs to store only information about which sectors are unallocated.” Ex.1017, 3:1-12. Ex.1003, ¶117.

Le further evidences that CoW was well-known. Le explains that it was known to “employ the copy-on-write technique to allow a virtual machine to modify its virtual disk without actually modifying its virtual disk file,” such that “[w]hen copy-on-write is enabled for a virtual disk, modifications to the file are stored in a separate file, sometimes called a redo log” that “specifies which sector locations in the original disk were written and contains the modified contents for those locations.” Ex.1017, 13:55-64. These redo logs “may also be ‘chained’ as a sequence of ‘delta’ disks, each of which records writes to the virtual disk since a most recent preceding checkpoint,” such that “[t]he first such delta disk thus records changes to

the initial state of the virtual disk; the second delta disk records writes after the first delta disk is checkpointed; and so on.” Ex.1017, 14:4-9. Ex.1003, ¶118.

Virtualization, an indirection technique that makes one thing seem like another, was also well-understood by the time of the ’282 Patent, and in the context of logical drives, storage systems, disk devices, drivers, and file systems. Ex.1003, ¶¶119-131 (citing Ex.1011 at 41, 52-53, 61-64).

It was also well-understood that any single device (*e.g.*, a hard disk (a type of non-volatile storage)) can be logically broken down into units called “partitions” that act as their own logical storage area or disk. Ex.1011, 15 (“**disk partition *n***. A logical compartment on a physical disk drive. A single disk might have two or more logical disk partitions, each of which would be referenced with a different disk drive name.”). A POSITA would have understood that in general, a “partition” is “1. A logically distinct portion of memory or a storage device that functions as though it were a physically separate unit.” Ex.1011, 47. Ex.1003, ¶¶59-63.

The cited references detailed below further demonstrate the claims of the ’282 Patent were well known and obvious before the ’282 Patent.

A. Birse (Ex.1005)

Birse relates to the provision of an OS in a net-booted environment using “a copy-on-write storage scheme in which modifications (writes) directed to the user system volume 930 are instead copied to the shadow system volume 930.” Ex.1005,

1:18-21, 14:56-61. A server in a network computer (“NC”) system maintains a copy of the OS that cannot be corrupted by ordinary users of the NC system. Additionally, the NC system may preserve user customizations (*e.g.*, preferences) by maintaining individual, user, storage areas. When an NC client boots from the network and accesses a stored copy of the OS from an NC server, the user’s preferences are dynamically merged with the system environment provided to the NC client. *Id.*, 2:56-65. Advantageously, because of these features, a user may login to any NC client on the network and have the same user experience. *Id.*, 2:65-3:3. Ex.1003, ¶¶173-83.

The NC system includes an NC client 150 (*e.g.*, personal computer) coupled to an NC server 170 via data communication line 140. Ex.1005, 4:65-5:8. NC server 170 can include one or more system volumes 174 (including a protected, read-only, master copy of the OS software), one or more application volumes 176, a user registry 178, non-persistent client-specific data 184 (temporary data storage), and persistent user data 186 (long-term storage for user information, *e.g.*, preferences, browser bookmarks, other desktop environment customizations). *Id.*, 5:9-29, 6:43-48.

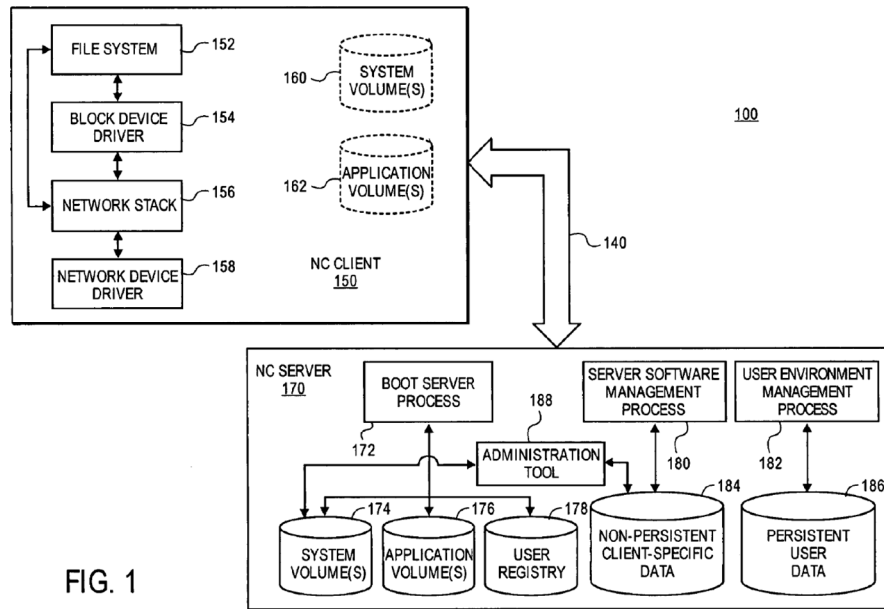


FIG. 1

Ex.1003, ¶¶184-87.

The NC server also includes a boot server process 172 (managing access to the system volumes 174 and the application volumes 176), a server software process 180 (managing access to the non-persistent client-specific data 184), and a user environment process 182 (maintaining the persistent user data 186). Ex.1005, 5:46-6:34. Ex.1003, ¶¶188-92.

The NC client also includes a file system 152 (logical structures and software routines to control access to and from local and remote storage media), a block device driver 154 (servicing file system read/write requests to the system volume 160 or shadow volumes, discussed below), a network stack 156 (used to access the data communication link 140), and a network device driver 158. Ex.1005, 7:6-60. Ex.1003, ¶¶193-96.

When an NC client boots from the network and accesses the operating system from an NC server, the user's preferences are dynamically merged with the system environment provided to the NC client. Ex.1005, 2:62-65. Birse uses one or more shadow volumes that are created from the NC client 150 and stored with the non-persistent client-specific data 184 for the purpose of storing user preferences and/or changes to the operating system. *Id.*, 7:41-46. Ex.1003, ¶¶197-202.

Specifically, Birse discloses an NC client/server system that includes a base/root image (Core System Volume 922) and at least one leaf/incremental snapshot (shadow system volume 930). Ex.1005, 14:24-30, 15:1-7. This is seen in Birse's Figure 9, reproduced below.

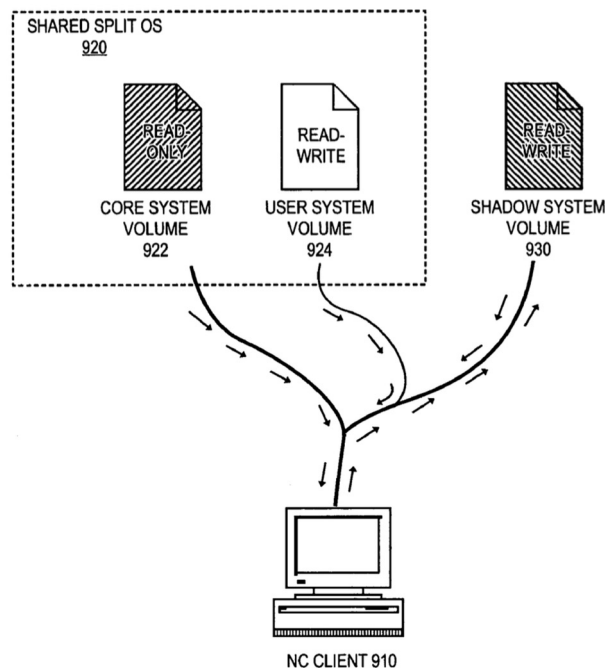


FIG. 9

Ex.1003, ¶197.

FIG. 9 illustrates a “SplitOS 920 of the NC server 170” that “contains a read-only core system volume image 922 and a read-write user system volume image 924.” Ex.1005, 14:24-30. Birse states that “[a]s indicated by the directional arrows along the connections between the core system volume 922, the user system volume 924, and the shadow system volume 930 and the NC client 910, data may be read from each of the core system volume 922, the user system volume 924, and the shadow system volume 930; however, data may only be written to the shadow system volume 930.” Ex.1005, 15:1-7. A POSITA would have understood Birse to be similar to managing snapshots as described above in the beginning of Section VI. Ex.1003, ¶¶198-99.

In Birse, “the shadow system volume 930 is used by the block device driver 154 of the NC client 150 to implement a ‘copy-on-write’ storage scheme in which modifications (writes) 60 directed to the user system volume 930 are instead copied to the shadow system volume 930.” Ex.1005, 14:57-61. A POSITA would thus have understood Birse to manage snapshots using CoW techniques. Ex.1003, ¶200.

“The block device driver 154 services file system read and write requests to the system volumes 160. In one embodiment, the block device driver 154 appears to the file system 152 to be a standard hard drive device driver. However, in reality, the block device driver 154 may be configured to service the read and write requests of

the file system 152 by accessing the NC server 170.” Ex.1005, 7:35-41. A POSITA would have understood Birse to include a (logical/virtual) device driver that resides between file systems and storage devices, and merges/unifies I/O requests from applications. This functionality is an integral part of snapshot storage systems. Ex.1003, ¶201.

Birse discloses that “the shadow system volume 930 is used by the block device driver 154 of the NC client 150 to implement a ‘copy-on-write’ storage scheme in which modifications (writes) 60 directed to the user system volume 930 are instead copied to the shadow system volume 930.” Ex.1005, 14:57-61. A POSITA would have understood that Birse’s block driver implements the functionality of managing snapshots including merging their content, using standard CoW techniques. Ex.1003, ¶202.

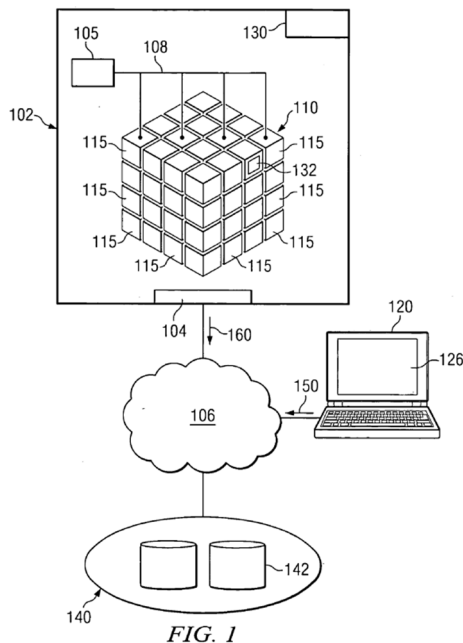
Thus, Birse discloses data structures and methods that a POSITA would have found functionally identical to the ’282’s UBD, especially in light of the ’282 patent’s description of what a UBD may be and how it reportedly works. Ex.1003, ¶203.

B. Ballew (Ex.1006)

Ballew relates to a “high performance computer system and method” in which a “high performance computing (HPC) system 100” is used for “executing software, applications, and processes.” Ex.1006, Title, ¶[0014]. Ballew’s Background section

describes that HPC systems were known and allowed for large tasks to be performed, such as by scientists and engineers for modeling, simulating, and analyzing complex physical or algorithmic phenomena,” and were usually arranged in the form of “HPC clusters” Ex. 1006, ¶[0002]. The ’282 Patent also discusses using HPC clusters. Ex.1001, 1:20-38 (“[o]ver the years, as the interne[t] has expanded and computers have multiplied, the need for clustered computing such as High Performance Computing (HPC) has increased”), 3:62-4:10. Ex.1003, ¶204.

See FIG. 1 of Ballew, below.



The system 100 offers “HPC functionality dynamically allocated among various computing nodes 115” of an HPC server 102 (multiple servers in a “server pool”), where the plurality of nodes 115 can “comprise any computer or processing device such as, for example, blades, general-purpose personal computers (PC),

Macintoshes, workstations, Unix-based computers, or any other suitable devices.”
Ex. 1006, ¶¶[0016]-[0017]. The system 100 can operate as a “clustered application server” or a “remote web server.” *Id.* Ballew also discloses that the HPC server 102 can include a grid 110 comprising the plurality of nodes 115, where the grid 110 “is a group of nodes 115 interconnected for increased processing power.” Ex. 1006, ¶¶[0016]-[0017]. FIGS. 2C and 2D, shown below, illustrate examples in which a grid 210 has virtual clusters 220 allocated within the grid.

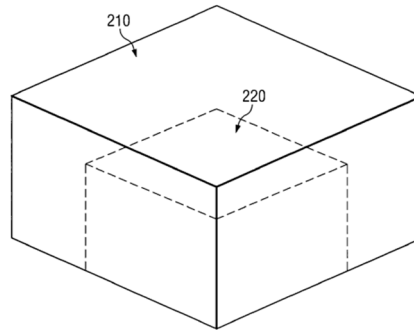


FIG. 2C

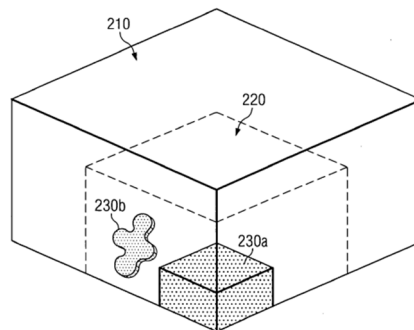


FIG. 2D

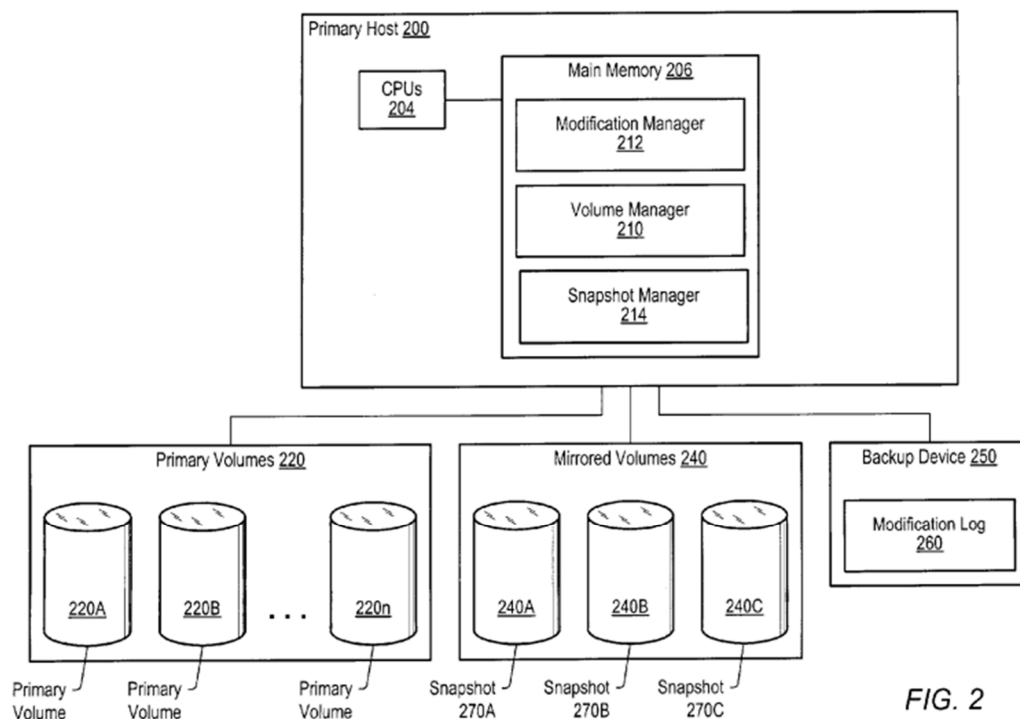
Ex.1003, ¶¶205-206.

When the HPC server 102 “is first initialized or booted” a “cluster management engine 130 may dynamically allocate various portions of grid 110 to one or more virtual clusters 220 based on, for example, predetermined policies,” and

can “also manage and provide a unique execution environment for each allocated node of virtual cluster 220.” Ex. 1006, ¶[0029]. Ballew explains that “[t]he execution environment may consist of the hostname, IP address, operating system, configured services, local and shared file systems, and a set of installed applications and data.” *Id.* Ex.1003, ¶207.

C. Wu (Ex.1007)

Wu relates to a “techniques for reconstructing deleted snapshot copies of a data volume” in which snapshots “corresponding to an original volume” are created at the “volume-level” by storing data blocks of the original volume in the snapshots. Ex.1007, 1:8-10, 4:51-60. See FIG. 2 of Wu, below.



Ex.1003, ¶208.

Wu describes “whenever the original volume is modified, a copy of the pre-modification value of the modified block may be stored (e.g., in a copy-on-write file)” and then, “[w]hen the snapshot is used (e.g., to make a backup), the virtual device may be read,” such that “[r]ead accesses to the virtual device targeting a block that was modified subsequent to the creation of the snapshot may be remapped to the stored copy of the pre-modified value of that block” to provide “a point-in-time copy of the original volume.” Ex.1007, 4:61-5:4. Ex.1003, ¶209.

Wu discloses using a modification log to track modifications to the original volume between any two snapshots, such that, “[e]ach time a modification is detected, modification manager 212 may log the time at which the modification occurred, the address(es) of the modified block(s), and the pre- and post-modification values of the modified block(s).” Ex.1007, 6:14-23. If a snapshot is deleted, the modifications stored in the modification log can be used to reconstruct the snapshot using another snapshot. Ex.1007, 7:26-29. For example, “[i]f the deleted snapshot was created after the existing snapshot, the snapshot manager 214 may modify the existing snapshot by sequentially applying modifications logged in the modification log(s) 260 between the time the existing snapshot was created and the time the deleted snapshot was created.” Ex.1007, 7:36-41. Ex.1003, ¶210.

D. Kleinschnitz (Ex.1008)

Kleinschnitz relates to tools and techniques for facilitating management of storage, software, and other computer resources using system images that can be stored remotely and deployed to the original imaged computer or other computers.

Ex.1008, Abstract, 1:8-10, 4:51-60. See FIG. 2 of Kleinschnitz, below.

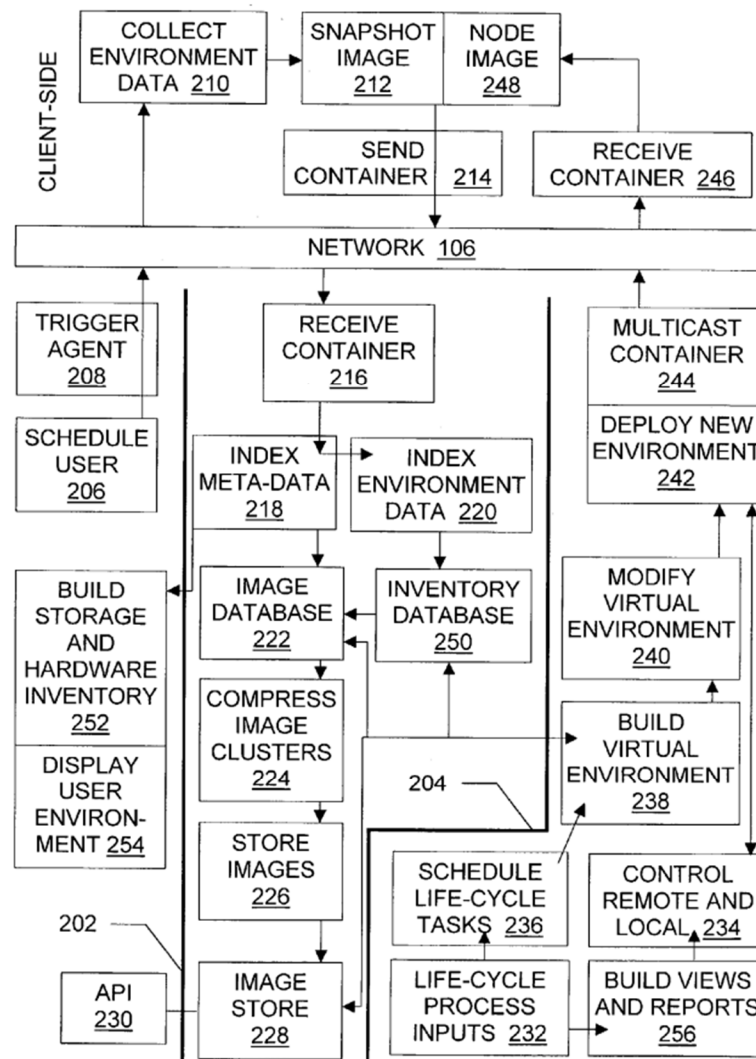


Fig. 2

Ex.1003, ¶211.

Kleinschnitz discloses using snapshots that include “incremental images” reflecting “the state of the computer’s storage” and storing the snapshots at another storage location. Ex.1008, [0003], [0035], [0059]. Kleinschnitz explains that “an incremental image 104 may be merged with a prior complete snapshot image 104 to form a complete and more recent snapshot image.” Ex.1008, [0071]. Kleinschnitz also discloses that storage disks can be partitioned, such as “by placing user data in one partition and system data and software in another partition.” Ex.1008, [0079]. Ex.1003, ¶212.

VII. GROUND 1: CLAIMS 1, 3-10, 13-16, 19-26, 29, 30 ARE OBVIOUS OVER BIRSE

A. Birse meets all claim elements.

Claims 1 and 15 recite substantially overlapping elements and are discussed together. *See* Ex.1010.

1. Claims 1 and 15

[1.0]/[15.0] “[1.0:A system][15.0:A method] for distributing an application environment comprising:”

Birse discloses a system and method for distributing an application environment (from NC server 170) to a plurality of compute nodes (NC clients 150, 812, 814, 816, 822, 824, 826): “[T]he NC clients [150] boot over the network by accessing a stored copy of the operating system from the NC server [170]. Additionally, when NC clients want to run an application, they access it from the

NC server.” Ex.1005, 4:56-60. *See also id.*, 2:51-3:14 (alternative embodiments providing data from NC server to NC clients to upgrade operating system, upgrade applications). This is illustrated in Birse’s Figure 1, reproduced below:

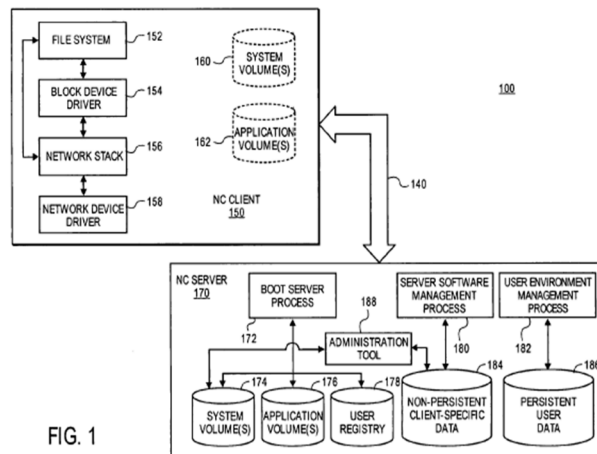


FIG. 1

FIG. 3 of Birse shows an example method of communication between an NC client 310 and an NC server 320:

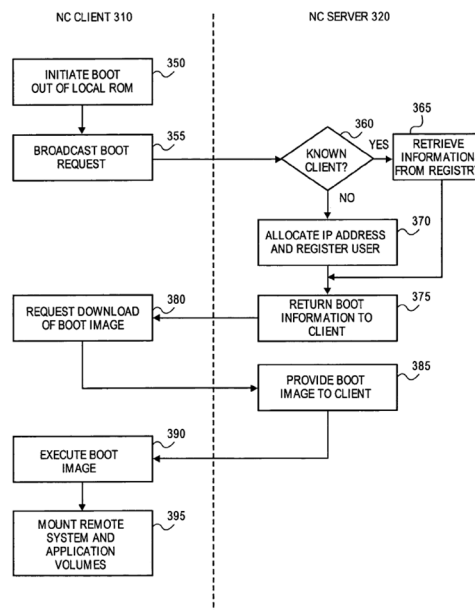


FIG. 3

Ex.1003, ¶¶216-17, 318-321.

**[1.1]/[15.1] “[1.1:a compute node comprising a computer system;
a first storage unit for] storing blocks of a root image of
[1.1:the][15.1:a] compute node[15.1: on a first storage unit, wherein
the compute node comprises a computer system],**

Birse relates to an “NC system” that includes “one or more NC clients,” (*i.e.*, “a compute node”), and “one or more NC servers” that “may provide access to a copy of the operating system to an NC client in response to a boot request.” Ex.1005, 4:31–40. “FIG. 8 is a simplified block diagram illustrating two NC systems 810 and 820 coupled via the Internet 805.” *Id.*, 14:8-9. Birse’s Figure 8 is reproduced below:

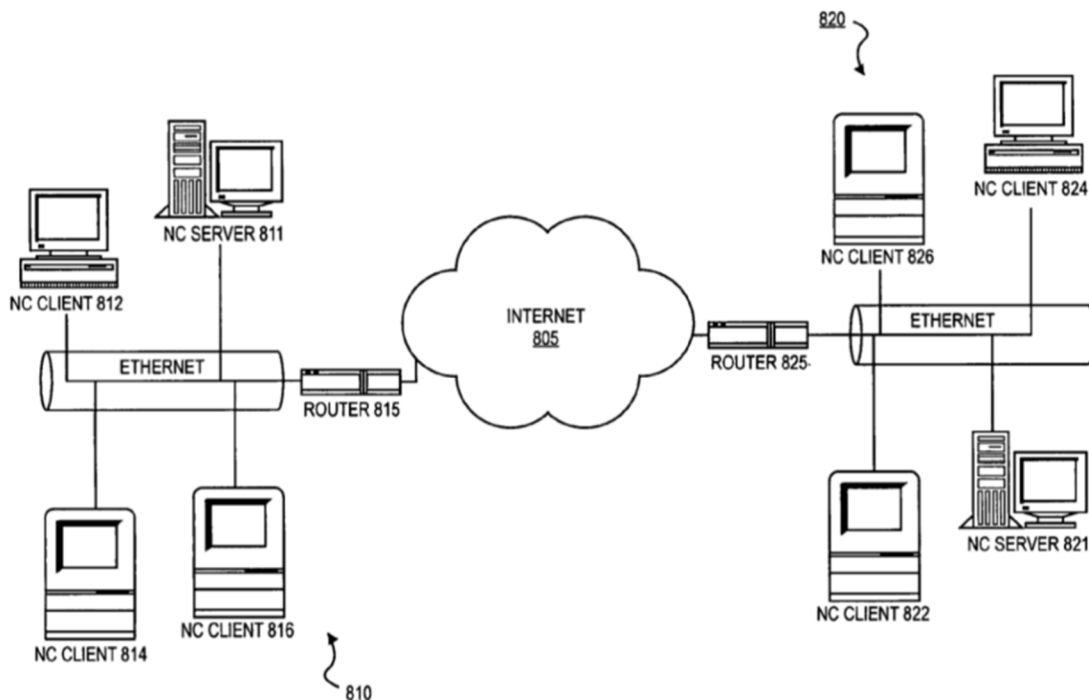
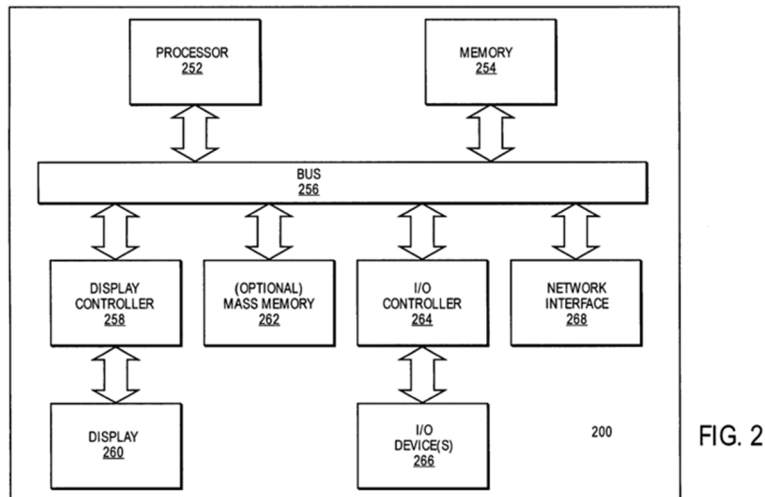


FIG. 8

Further, NC server 170 and NC clients 150 are “network computer[s] (NC)” (Ex.1005, 1:57-58) and are illustrated by exemplary digital processing system 200 of FIG. 2. As shown in FIG. 2 of Birse, and described at 7:63-8:35, the compute nodes comprise a computer system (“the digital processing system 200 shown in FIG. 2 may be used as an NC client, an NC server, or other server *computer system*”).



Ex.1003, ¶¶218-219.

Birse further discloses storing blocks of a root image (system volumes 174, 650, 660, 922, 924 on NC server 170) of a compute node on a first storage unit (one or more hard drives 610, logically separate in NC folders/directories 630, 640): “The NC server 170 includes one or more system volumes 174.... The system volumes 174 include a protected, read-only, master copy of the operating system software.”

Ex.1005, 5:10-14. Ex.1003, ¶220.

Further, “the NC server 170 contains a read-only core system volume image 922 and a read-write user system volume image 924 The core system volume 922 preferably contains those parts of the system that do not need to be written back to during system operation. One goal of the core system volume 922 is to provide all the system components that are mandatory for system operation.... The user system volume 924 contains all the user-configurable system components, including preferences and all the system additions installed by application software, such as application-installed extensions and libraries.” Ex.1005, 14:27-34, 14:41-44.

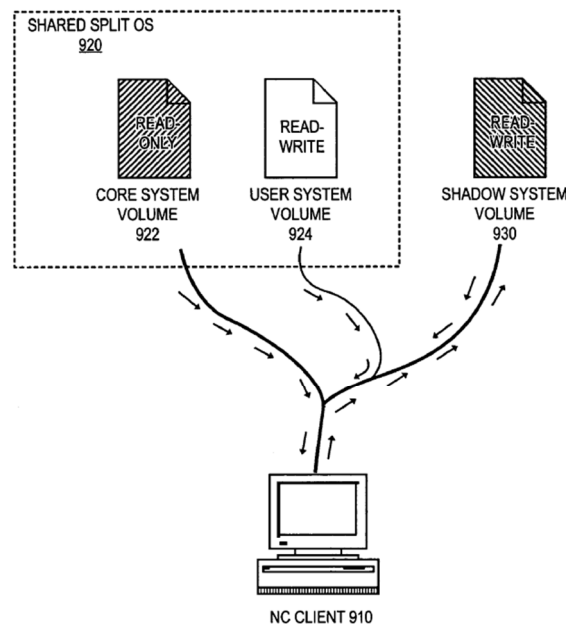


FIG. 9

As shown above by the arrows of FIG. 9, and as further explained below, the NC client reads data from the core system volume 922 and user system volume 924,

and writes requested by the NC client are directed to the shadow system volume 930.
Ex.1005, 14:56-67. The “system volumes 174...may be independently distributed
across multiple server computer systems.” *Id.*, 6:59-63. Ex.1003, ¶¶221-222.

FIG. 6 of Birse “is a simplified block diagram illustrating the layout of an
exemplary hierarchical directory structure that may be used by an NC server [170]
according to one embodiment.... [A] hard drive 610 of the NC server 170... includes
an NC folder (directory) 620. Within the NC folder 620 are an NC admin folder 630
and an NC shared data folder 640.” Ex.1005, 12:30-38. Birse’s Figure 6 is
reproduced below:

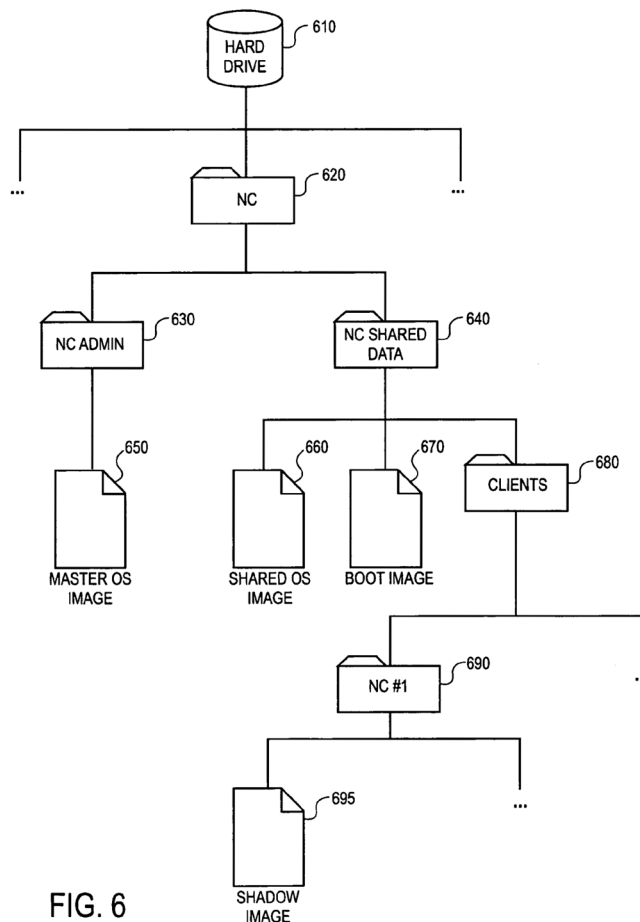


FIG. 6

As shown above, Birse discloses storing blocks of the root image on a first storage unit (specific folders/directories on hard drive 610). That is, Birse discloses storing in an NC folder 620 (a first storage unit) on hard drive 610 the master (read-only) OS image 650 (in folder 630) and the shared OS image 660 (in folder 640). “The shared operating system image 660 is a read-write version of the read-only master operating system image 650 and may be mounted by the NC client 150.” *Id.*, 12:44–47. Ex.1003, ¶¶223-224.

System volumes (a.k.a. system volumes 174, 650, 660, 922, 924, 1010) are stored as blocks as shown in Fig. 10, reproduced below:

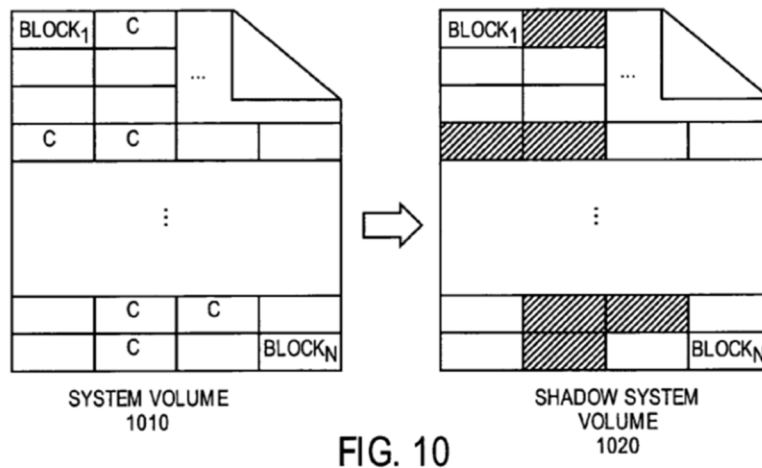


FIG. 10

Ex.1003, ¶225.

Additionally, although Birse discloses using both a master OS image 650 / user system volume 922 and a shared OS image 660 / user system volume 924, it would be apparent to a POSITA that these components meet the claimed root image of the '282 Patent. In Birse, the shared OS image 660 / user system volume 924 is

located in a storage location accessible to the Birse's block device driver 154 and is simply "a read-write version of the read-only master operating system image 650" to ensure that the original master OS image 650 is stored securely and to avoid any attempt to "delete or move items that are essential to system operation." Ex.1005, 14:35-40. However, a POSITA would have found it obvious that the master OS image 650 / user system volume 922 and the shared OS image 660 / user system volume 924 operate in the same way as the claimed root image to store the original, unmodified, blocks of the system volume because an OS (kernel) image alone is only part of what a computer system needs to run: it also needs a system volume with an assortment of critical administrator/user utilities. Ex.1003, ¶226-27, 322.

[1.2]/[15.2] "wherein the first storage unit comprises a first non-volatile memory, wherein the root image comprises a computer program, wherein the blocks comprise sections of data, and wherein a file of the root image comprises at least one block;"

Birse discloses that the first storage unit is a first non-volatile memory. As described above, the system volume in Birse (the root image) is stored within a folder/directory on a "hard drive 610...or other mass storage device." Ex.1005, 12:34-42. A POSITA would understand that a hard drive or other mass storage device is non-volatile memory. Further, Birse specifically discloses that hard drives/disks are non-volatile memory: "a non-volatile mass memory (e.g., a hard disk)." Ex.1005, 16:1-7. Ex.1003, ¶¶228-29.

Birse also discloses that the root image comprises a computer program. As described above, Birse specifically discloses that the system volumes comprise an operating system (e.g. operating system images 650, 660). Ex.1005, 12:44–47. Birse specifically discloses that “[a]n ‘operating system’ refers to a **program** that manages a computers internal functions and provides a means to control the computer's operations and file system.” Ex.1005, 4:7-9. Ex.1003, ¶230.

Birse also discloses that the blocks of the root image comprise sections of data. First, it is noted that the word “section” only appears in the claims of the '282 Patent. As explained above, Birse’s system volumes (a.k.a. system volumes 174, 650, 660, 922, 924, 1010) are stored as blocks (sections of data) as shown in Fig. 10, reproduced below:

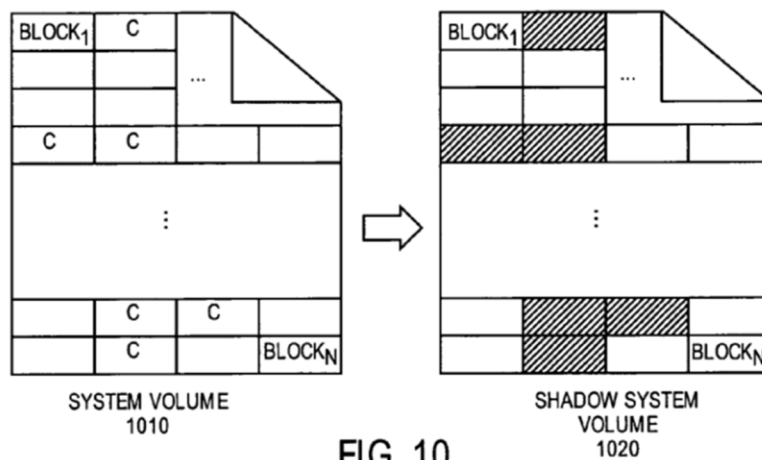


FIG. 10

Ex.1003, ¶231.

Birse further describes embodiments in which data can be written in bands, where bands are a predetermined number of blocks. These bands of blocks include

sections of data, e.g., bands of “128K (256 512-byte blocks).” Ex.1005, 15:20-32. Ex.1003, ¶232.

Birse further discloses or suggests that a file comprises at least one block. As described above, the system volumes are operating system images. Operating systems include files and it would be obvious that, depending on the size of a file of the operating system stored in the operating system volumes of Birse, that one file could comprise at least one block of the system volume. Ex.1003, ¶¶233. As explained in Section VI above, it was well-known prior to the '282 Patent that the content of a file can be stored in one or more blocks or sectors. *See, e.g.*, Ex.1017, 1:16-20. Ex.1003, ¶¶117-118, 233-34, 323.

[1.3]/[15.3] “[1.3:a second storage unit for] storing a leaf image[1.3:, the leaf image] comprising new data blocks and changes to the blocks of the root image[15.3: on a second storage unit], wherein the second storage unit comprises a second non-volatile memory”

As explained in Section VI.A. above, Birse discloses using a copy-on-write technique to store a leaf image (shadow volume/image 930, 695) for respective compute nodes (NC clients 150, 812, 814, 816, 822, 824, 826) on a second storage unit (NC client folders 690 #1...n on hard drive 610), where the leaf image includes new data blocks of the root image (write requests for new data) and changes to the blocks of the root image (write request for changes to operating system data): “In

this example, the NC clients are numbered from 1 to N and the clients folder 680 includes a folder for each NC client. For example, an NC #1 folder 690 is an area for storing client-specific information, such as a shadow image 695, corresponding to NC client #1.” Ex.1005, 12:54–59.

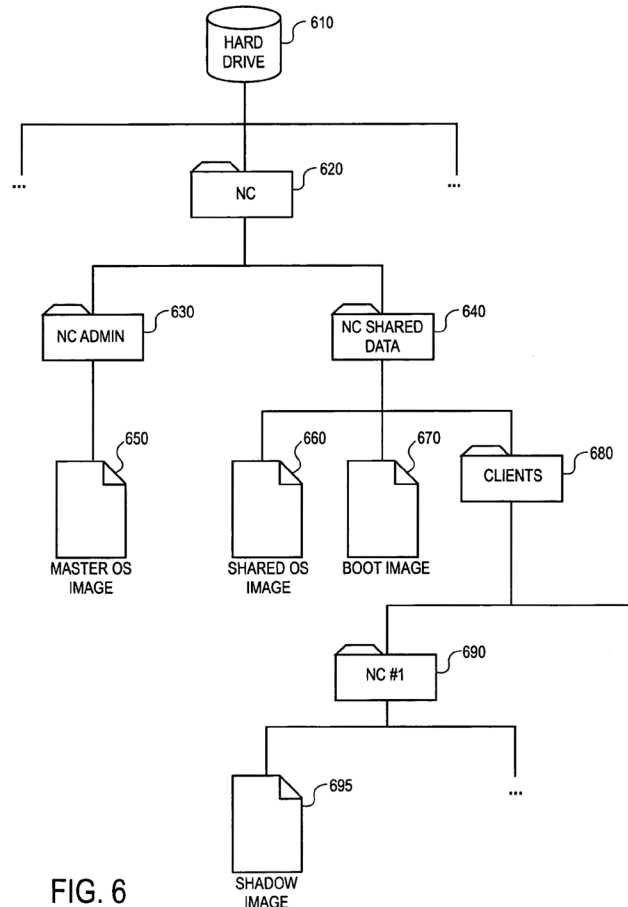


FIG. 6

Ex.1003, ¶¶173-203, 235.

Birse discloses that the second storage unit comprises a second non-volatile memory. First, the '282 Patent describes that “[t]he first storage unit 140 and second storage units 150a-n may each be contained on separate physical storage devices, on separate logical spaces on the same storage device, or any combination thereof.”

Ex.1001, 4:17-22. In embodiments of Birse, the first storage unit and the second storage units are in separate logical spaces on the same storage device, that is, the first storage unit is in NC folders/directories 630, 640 (containing the master OS image and shared OS image (root image)) and the second storage unit is in NC folders/directories 680, 690 (containing the shadow image (leaf image)). Ex.1005, 12:30-64. To the extent the Patent Owner asserts that the first non-volatile memory and the second non-volatile memory must be separate physical storage devices (e.g., separate hard drives), it would be obvious to do so based on the disclosure of Birse. Particularly, Birse discloses that I/O device 266 shown in FIG. 2 can include multiple “disk drives” (Ex.1005, 8:32-35), memory used for storage can naturally “compromise one or more disks” (*id.*, 15:58-62), and that, while Birse describes “a single NC server 170” server processes and data including the system volumes 174 and client-specific data 184 (where the client-specific data 184 can include shadow volumes (leaf images)) can be “distributed across multiple server computer systems.” *Id.*, 6:59-63, 7:41-46. Ex.1003, ¶¶236-37.

Regarding the leaf image, Birse discloses: “block device driver 154 services file system read and write requests to the system volumes 160. In one embodiment, the block device driver 154 appears to the file system 152 to be a standard hard drive device driver ... [but is] configured to service the read and write requests of the file system 152 by accessing the NC server 170. As will be discussed further below, in

one embodiment, one or more shadow volumes [695], may be created for the NC client 150 and stored with the non-persistent client-specific data 184 for the purpose of storing user preferences and/or ***changes to the operating system***. Therefore, some mechanism is needed to direct reads and writes to the appropriate volume, e.g., the shadow volumes or the system volumes 174. According to one embodiment, the block device driver 154 has knowledge of the separate shadow volumes and system volumes 174 and redirects read and write requests to the appropriate volume.” Ex.1005, 7:35-53; *see also id.*, 11:30-32. Ex.1003, ¶238.

FIG. 4 of Birse “is a flow diagram illustrating how NC client [150] write requests directed to the system volumes 160 are handled.... [A]t step 410, the file system 152 generates a write request directed to the system volumes 160. The write request is received by the block device driver 154 and redirected by the block device driver 154 at step 420 by translating it into a write request directed to the user’s shadow volume on the NC server 170. At step 430, the NC server 170 stores information associated with the write request in the shadow volume associated with the NC client 150. In this manner, the file system 152 need not be aware that attempted modifications to the system volumes 160 are recorded instead in a client-specific shadow volume residing on the NC server 170.” Ex.1005, 10:63-65, 11:6-17. Birse’s Figure 4 is reproduced below:

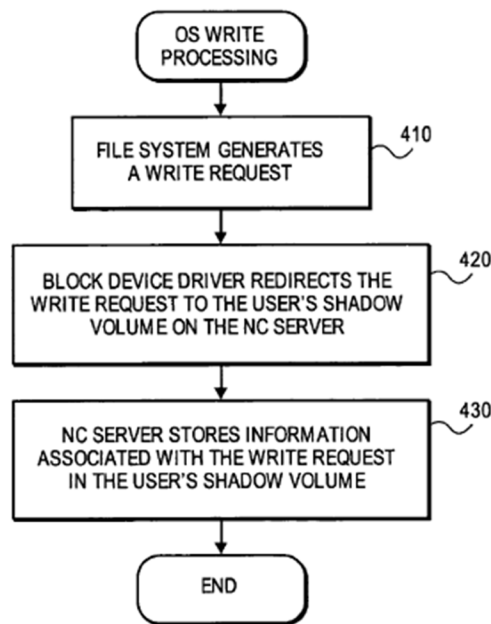


FIG. 4

Ex.1003, ¶239.

Embodiments of Birse disclose that the “shadow volume contains only those portions of the operating system that have been modified by the user rather than a complete copy of the operating system as modified.” Ex.1005, 11:49-52. Another portion of Birse states that “the shadow image 695 preferably contains only portions of the shared operation system image 660 that have been modified by the user.” *Id.*, 12:59-61.

Similarly, “at step 510, the file system 152 generates a read request directed to the system volumes 160,” which “is received by the block device driver 154” at step 520:

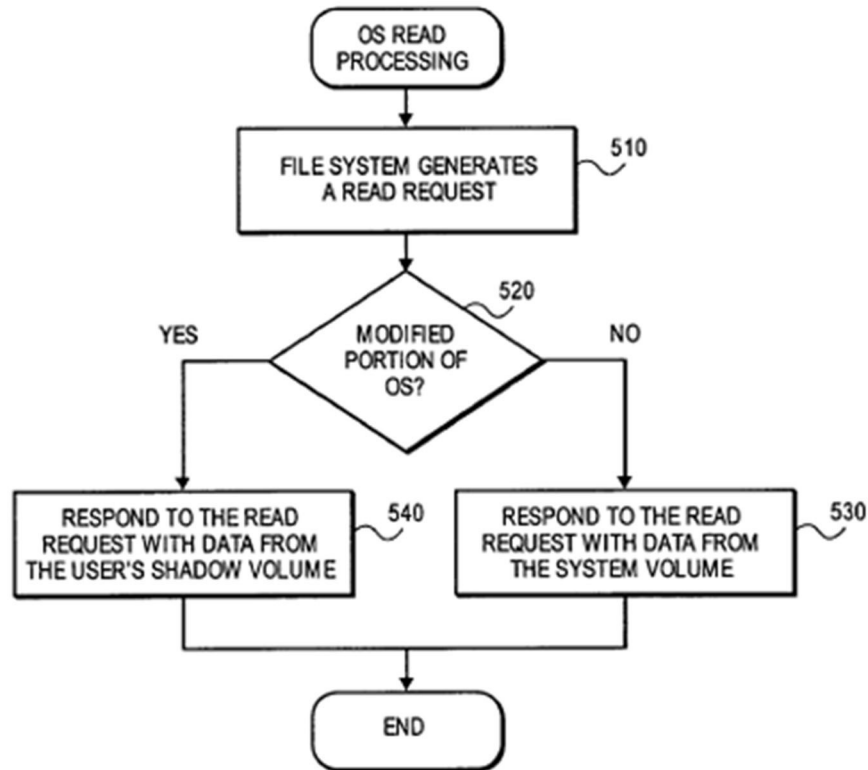


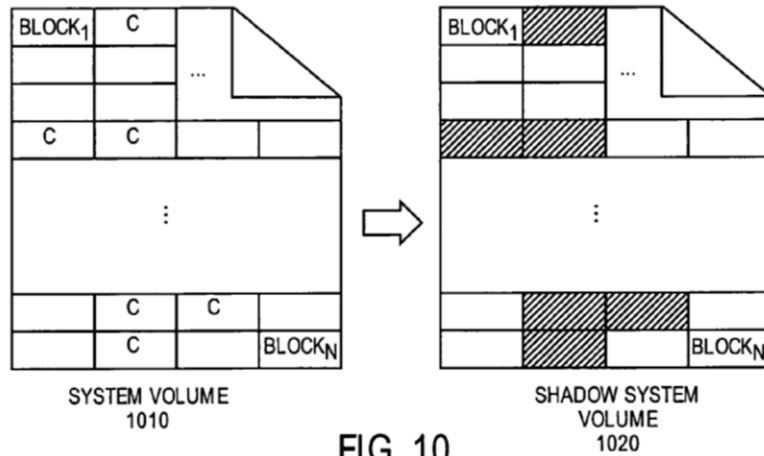
FIG. 5

Id., Fig. 5. Rather than reading directly from the system volume, “a determination is made whether the read request specifies a portion of the operating system that has been modified by the user or whether the read request specifies a portion of the operating system that remains unchanged.” *Id.*, 11:55–62. If the read request “corresponds to a portion of the operating system that has not been modified by the

user, the processing continues with step 530,” wherein “the block device driver directs the NC server 170 to retrieve the information associated with the read request from the system volumes 174.” *Id.*, 11:62–12:3. Alternatively, “if the read request corresponds to a portion of the operating system that has been previously modified by the user, then processing continues with step 540,” wherein “the block device driver directs the NC server 170 to retrieve the information associated with the read request from the user’s shadow volume.” *Id.*, 12:4–6. Ex.1003, ¶¶240-42.

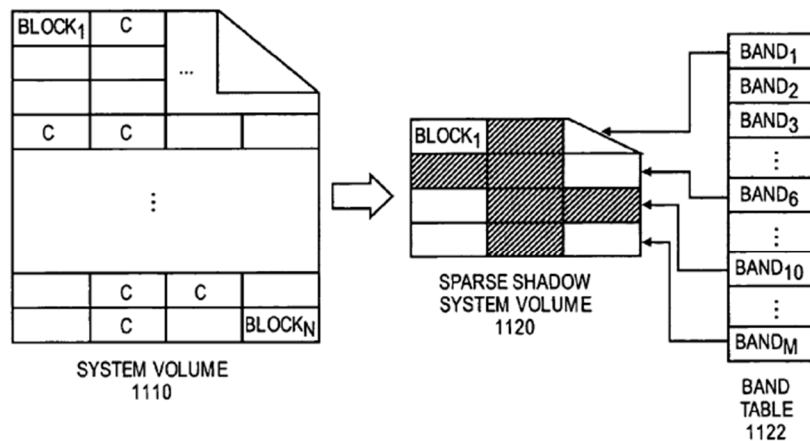
The NC server “creates a shadow system volume 930 for each connected NC client. The shadow system volume 930 shadows the user system volume 924 by storing modifications that are made to the user system volume 924.” Ex.1005, 14:49-53. Moreover, “... the shadow system volume 930 is used by the block device driver 154 of the NC client 150 to implement a ‘copy-on-write’ storage scheme in which modifications (writes) directed to the user system volume 930 are instead copied to the shadow system volume 930.” *Id.*, 14:56-61. Ex.1003, ¶¶243-44.

These modifications in the shadow system volumes of Birse are stored as blocks, as shown for example in Fig. 10, reproduced below:



Ex.1003, ¶245.

Birse further describes embodiments in which data can be written in bands in a sparse shadow system volume, as shown in FIG. 11 below.



Ex.1005, 15:20-32. The sparse shadow system volume 1120 “is written in band increments,” where a band “is a predetermined number of blocks,” such that “the disk space used by the shadow volume 1120 will consist only of the actual data written (in band increments).” *Id.* Ex.1003, ¶¶246-47.

This banding feature provides an “inherent one-to-one block mapping in copy-on-write images” which Birse discloses can also “be replaced with a table or other data structure that maps a band of the primary file to a corresponding band in the shadow file.” Ex.1005, 15:32-37. Logic may be “included in the block device driver 154, for example, to break up requests into the largest contiguous chunks possible” so that “both reads and writes can span non-contiguous band boundaries.” Ex.1005, 15:37-40. Banding as described in Birse “reduce[s] disk space requirements of the NC client shadow system images” and “the reduction is achieved by only storing those bands which have actually been written to. Ex.1005, 15:41-45. Birse describes that the “an exemplary band table 1122 is a simple array allocated at initialization” and entries in the table are assigned BAND_NONE until the corresponding band is written to, at which point the next available band in the shadow file is assigned.” Ex.1005, 15:45-49. Further, “in order to encourage contiguous blocks on the NC server's file system, in one embodiment, if bands are allocated during a write, the shadow file size is increased to the appropriate number of bands before the write request is sent.” Ex.1005, 15:45-52. Ex.1003, ¶¶248-50, 324.

[1.4] “a union block device for interfacing between the compute node and the first and second storage units to distribute the application environment to the compute node, wherein the union block device

comprises a driver,” [15.5] “delivering the application environment to the compute node;”

Birse teaches the use of a block device *driver* 154 that redirects read and write requests to/from a user’s shadow volume as necessary. Ex.1005, 7:35-53. Accordingly, it would have been obvious to a POSITA that the block device driver for each NC client is “a union block device for interfacing between the compute node and the first and second storage units...wherein the union block device comprises a driver,” as required by this claim.” Furthermore, the block device driver, system volumes, and shadow volumes “distribute the application environment to the compute node” as required by claim 1, and thus also “deliver[] the application environment to the compute node,” as required by claim 15. Ex.1003, ¶¶251-52.

The ’282 patent explains that, “[i]n a branching store file system, a read-only base image (or ‘root’ image) of the application environment is created,” which is “accessible by all compute nodes in the cluster,” and that “[c]hanges made by a compute node to the root image are stored in a ‘leaf’ image unique to that compute node.” Ex.1001, 1:62-67.

The Birse system operates the same way: the system volumes include “a protected, read-only, master copy of the operating system software,” and a “shadow volume contains only those portions of the operating system that have been modified by the user.” Ex.1005, 5:13–14, 11:49–52. The block device driver

redirects read and write requests between the system volumes and shadow volumes in a way that is transparent to the file system— in one embodiment, “the block device driver 154 appears to the file system 152 to be a standard hard drive device driver,” and “the file system 152 need not be aware of the particular mechanism used to coordinate user modifications with the original operating system as contained in the system volumes.” *Id.*, 7:35–38, 11:14–17. Ex.1003, ¶¶252-54, 326.

It should be noted here that the '282 Patent does not define what a UBD is, but does describe that UBDs may be low-level drivers. The '282 Patent states:

In one embodiment, UBDs 130*a-n* are effectively low-level drivers that operate as an interface between the first and second storage devices and the file system of each compute node 120*a-n*. ... Because UBDs 130*a-n* operate below the file system, they are concerned merely with the blocks of data themselves, rather than files they form.

Ex.1001, 4:42-51. Many storage drivers reside below file systems. Ex.1003, ¶157.

In the '282 Patent, UBDs are said to “determine what leaf image (from the appropriate second storage unit 150) is needed for portions of the application environment that their respective compute nodes 120*a-n* have changed. UBDs 130*a-n* also locate the portions of the application environment that are not changed by their respective compute nodes 120*a-n*. These portions may reside in the root image. There may also be intermediate images (not depicted in FIG. 1) comprising versions of the root image from which a compute node’s instance of the application environment is derived.” Ex.1001 ('282 Patent), 4:52-67. A POSITA would have

understood that this is no more than common traversal of copy-on-write snapshots. Ex.1003, ¶158.

Moreover, “UBDs 130*a-n* may also modify the leaf image in response to their respective compute node’s access to its instance of the application environment. For example, upon receiving a write request from their respective compute nodes for a sector X, the UBDs 130*a-n* will create an appropriate persistent mapping for sector X and then write sector X onto their respective second storage units 150*a-n*, where sector X can then be modified.” Ex.1001, 5:1-11. Again, a POSITA would have understood that the UBD functionality disclosed here is nothing more than typical copy-on-write (CoW) like that used in Birse. Ex.1003, ¶159.

Lastly, UBDs logically merge different root and leaf images, which are nothing more than snapshots. Thus “Step 330 is merging the blocks of the root image with the blocks of the leaf image to create the application environment. Once the application environment has been created, it will appear to the compute node as one cohesive image rather than a base image plus its additions, deletions, and modifications. To the compute node, it appears that it has access to its own unique version of an application. However, a separate and complete boot image is not actually stored for the compute node.” Ex.1001, 6:8-16. A POSITA would have recognized this merging functionality to be no different than what prior art snapshot systems already disclosed. Ex.1003, ¶160.

[1.5]/[15.4] “[1.5:wherein the union block device creates the application environment by] merging the blocks of the root image stored on the first storage unit with the blocks of the leaf image stored on the second storage unit[15.4: to create the application environment]”

The '282 patent explains that, “[a] filter operates between the compute nodes and the file system(s), which merges the changes recorded on the leaf images with the root images and delivers the result to the appropriate compute node,” which appears to a compute node as “its own unique and cohesive instance of the application environment.” Ex. 1001, 1:67-2:5.

The Birse system operates the same way: the block device drivers (*i.e.*, “union block devices”) “are configured to create said application environments by merging the blocks of said root image,” that is, data from the system volumes, “with the blocks of respective leaf images,” that is, the shadow volumes. Ex.1003, ¶¶255-56.

Birse discloses merging the blocks of the root image stored on the first storage unit (blocks of the operating system software stored on system volumes 174, 650, 660, 922, 924) with the blocks of the leaf image (695, 930) stored on the second storage unit (NC client folders 690 #1...n) to create the application environment (dynamically merged system environment): “In one embodiment, the shadow system volume 930 is not visible to the user. Consequently, the user system volume 924 will appear to be the location where all writes are directed. For example, if a

user modifies a preference, that preference will appear to be written to the preference file located in the preferences folder on the user system volume 924.” Ex.1005, 14:61-67.

When “an NC client [150] boots from the network and accesses the operating system from an NC server [170], the user’s preferences are dynamically merged with the system environment provided to the NC client [150].” *Id.*, 9:2-5. Also, “[a]dvantageously, since the user’s desktop preferences and other customized settings are all preserved from session to session and supplied to the NC client [150] as it boots from the network, the user may login to any NC client [150] on the network and have the same user experience.” *Id.*, 9:5-10.

“Referring now to Fig. 5, NC client [150] read request processing will now be described according to one embodiment of the present invention. ... At step 510, the file system 152 generates a read request directed to the system volumes 160.” At “step 520, the read request is received by the block device driver 154 and a determination is made whether the read request specifies a portion of the operating system that has been modified by the user or whether the read request specifies a portion of the operating system that remains unchanged.” If “the read request corresponds to a portion of the operating system that has not been modified by the user, the processing continues with step 530.” However, “if the read request corresponds to a portion of the operating system that has been previously modified

by the user, then processing continues with step 540.” At “step 530, the block device driver directs the NC server 170 to retrieve the information associated with the read request from the system volumes 174.” At “step 540, the block device driver directs the NC server 170 to retrieve the information associated with the read request from the user’s shadow volume.” Thus, “[i]n this manner, the file system 152 need not be aware of the particular mechanism used to coordinate user modifications with the original operating system as contained in the system volumes 174.” Ex.1005, 11:39-41, 11:55-12:10. Birse’s Figure 5 is reproduced below:

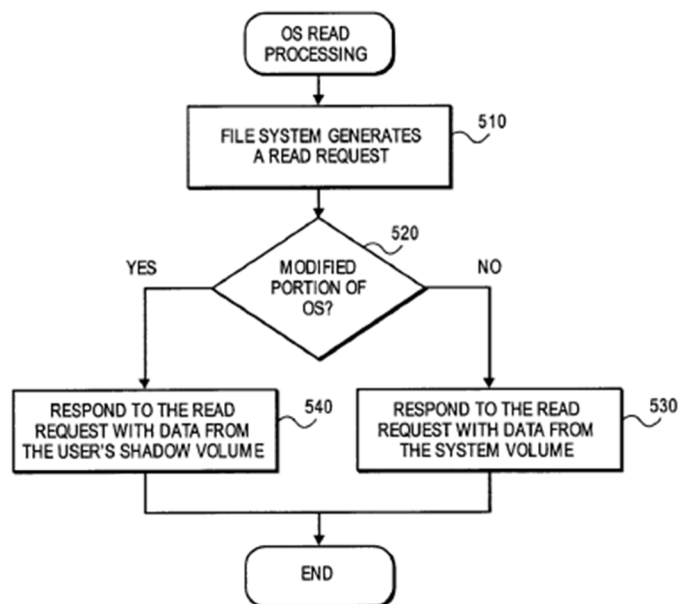
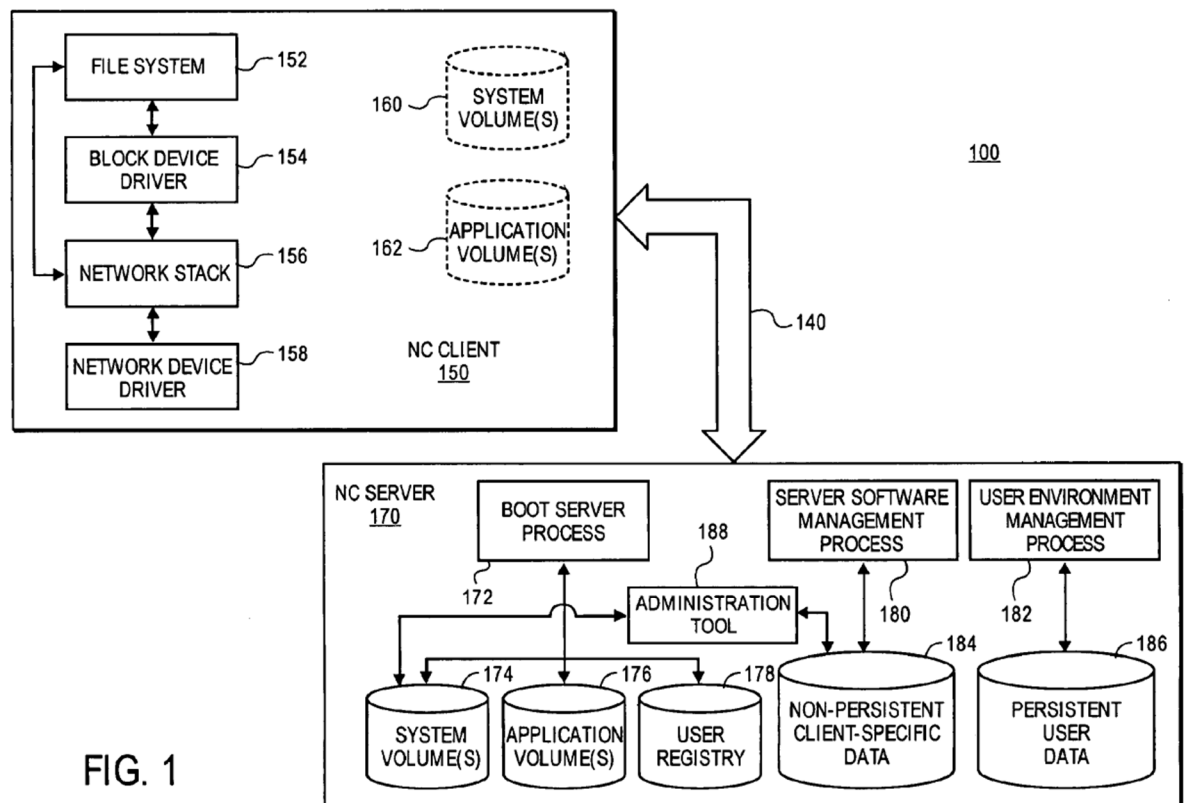


FIG. 5

Ex.1003, ¶¶257-60, 325.

[1.6] “the union block device comprises a low-level driver for interfacing between the first and second storage units and the file system of the compute node;”

As described above, Birse teaches the use of a block device *driver* that “keep[s] track of which portions of the operating system ... have been modified by the user” and redirects file system read and write requests between the system volumes and shadow volumes as appropriate. Ex.1005, 12:11-15. See FIG. 1 of Birse:



The “block device driver 154 appears to the file system 152 [of NC client 150] to be a standard hard drive device driver.” *Id.*, 7:35–38. Additionally, Birse specifies

that the file system and block device driver operates “as if the contents of the system volumes 160 are, in fact, contained on a local hard drive and the file systems read and write requests are redirected at a *lower level* of the operating system.” *Id.*, 7:27-31. Ex.1003, ¶¶261-63.

[1.7]/[15.6] “[1.7:and the union block device,][15.6:wherein the modifying comprises:] upon receiving a write request from the compute node for a sector X, [1.7:creates][15.6:creating] an appropriate persistent mapping for sector X”

As an initial matter, a POSITA would have understood that any storage system that maintains files, executables, CoW snapshots, and especially an operating system image, *must* persistently preserve the data blocks associated with those. In other words, data is persistent on non-volatile devices (*e.g.*, hard disks). If data was not persisted, for example maintained only in volatile RAM memory, all data (*e.g.*, user files and operating system images) would be lost upon system reboot or power outage. Additionally, for any storage snapshot system or system imaging system that stores snapshots or system images persistently, there must be a mapping of the original vs. modified blocks or it would not be possible to collapse/merge, remove, take, and/or restore snapshots or images. Ex.1003, ¶264. Furthermore, as explained above in Section VI above, it was well-known prior to the '282 Patent that sectors and blocks refer to the same subject matter and that sector-based image formats

describe the state of a disk at the sector, or block, level. *See, e.g.*, Ex.1017, 1:16-20. Ex.1003, ¶¶117-118, 264.

The only description associated with this claim language in the '282 Patent states:

UBDs 130a-n may also modify the leaf image in response to their respective compute node's access to its instance of the application environment. For example, upon receiving a write request from their respective compute nodes for a sector X, the UBDs 130a-n will create an appropriate persistent mapping for sector X and then write sector X onto their respective second storage units 150a-n, where sector X can then be modified. It will be appreciated that the data block being modified may already exist in the leaf image, in which case it does not need to be mapped and copied from the root image before modification.

Ex.1001, 5:1-11.

The system of Birse meets these claim elements. In Birse, “one or more shadow volumes [695], may be created for the NC client 150.” The “block device driver 154 has knowledge of the separate shadow volumes and system volumes 174 and redirects read and write requests to the appropriate volume.” Ex.1005, 7:35-53; *see also id.*, 11:30-32. Birse further explains that “the file system 152 generates a write request directed to the system volumes 160,” the “write request is received by the block device driver 154 and redirected by the block device driver 154 at step 420 by translating it into a write request directed to the user’s shadow volume on the NC

server 170,” and “the NC server 170 stores information associated with the write request in the shadow volume associated with the NC client 150.” *Id.*, 10:63-65, 11:6-17. As shown for example in FIG. 10 of Birse, blocks of the shadow volume (leaf image) that are modified in response to a write request (a sector X of the shadow volume) are persistently mapped (a “one-to-one block mapping in copy-on-write images”) and correspond to associated blocks of the original system volume (root image).

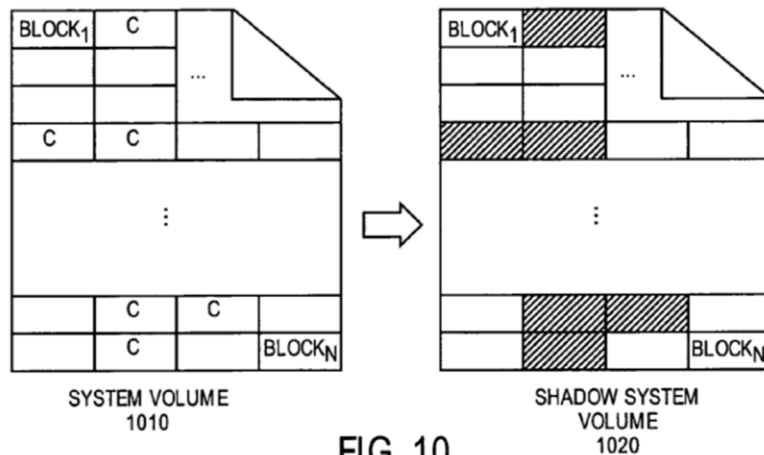


FIG. 10

Ex.1003, ¶¶265-66.

Birse also discloses an embodiment in which a “table or other data structure...maps a band of the primary file to a corresponding band in the shadow file.” Ex.1005, 10:63-65, 11:6-17, 15:33-37. Specifically, Birse describes embodiments in which data can be written in bands in a sparse shadow system volume, as shown in FIG. 11 below.

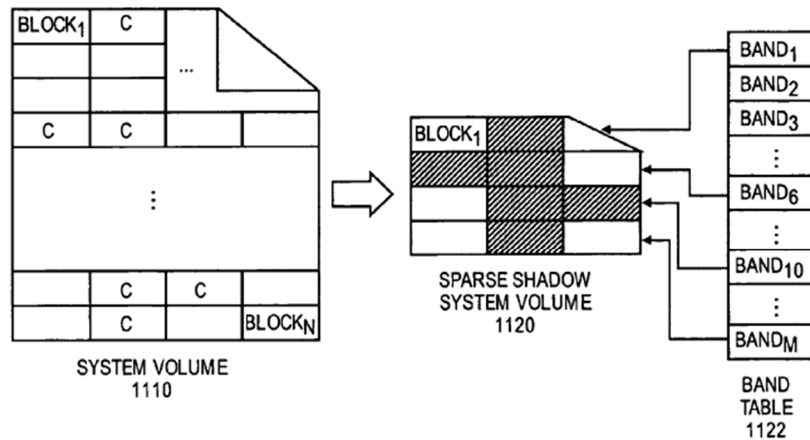


FIG. 11

Ex.1005, 15:20-32. The sparse shadow system volume 1120 “is written in band increments,” where a band “is a predetermined number of blocks,” such that “the disk space used by the shadow volume 1120 will consist only of the actual data written (in band increments).” *Id.* Ex.1003, ¶¶267-68.

This feature provides an “inherent one-to-one block mapping in copy-on-write images” which Birse discloses can also “be replaced with a table or other data structure that maps a band of the primary file to a corresponding band in the shadow file.” Ex.1005, 15:32-37. Logic may be “included in the block device driver 154, for example, to break up requests into the largest contiguous chunks possible” so that “both reads and writes can span non-contiguous band boundaries.” Ex.1005, 15:37-40. Banding as described in Birse “reduce[s] disk space requirements of the NC client shadow system images” and “the reduction is achieved by only storing those bands which have actually been written to. Ex.1005, 15:41-45. Birse describes that

“an exemplary band table 1122 is a simple array allocated at initialization” and entries in the table are assigned BAND_NONE until the corresponding band is written to, at which point the next available band in the shadow file is assigned.” Ex.1005, 15:45-49. Further, “in order to encourage contiguous blocks on the NC server's file system, in one embodiment, if bands are allocated during a write, the shadow file size is increased to the appropriate number of bands before the write request is sent.” Ex.1005, 15:45-52. Ex.1003, ¶¶269-70. As explained in Section VI above, it was well-known prior to the '282 Patent to use sparse image files like that in Birse to provide a sector-level representation of a volume or disk. *See, e.g.*, Ex.1017, 3:1-12. Ex.1003, ¶¶117-118, 271-72, 327.

[15.7] “writing sector X on the second storage unit.”

In Birse, writes to shadow volumes (and writes thus to sectors and/or blocks of the shadow volumes) are written on the second storage unit. As explained, the shadow volumes (and modifications/writes thereto) are on a second storage unit (e.g., NC client folders 690 #1...n on hard drive 610). Ex.1005, 12:42-63. As described above, Birse discloses persistently mapping sectors for writing blocks of its shadow volumes/images, where blocks of the shadow volume (leaf image) that are modified in response to a write request (a sector X of the shadow volume) are persistently mapped (a “one-to-one block mapping in copy-on-write images”) and

correspond to associated blocks of the original system volume (root image).
Ex.1005, 10:63-65, 11:6-17, 15:20-52. Ex.1003, ¶¶327-331.

2. Claim 16

[16] “modifying the leaf image in response to the compute node's access to the application environment.”

As explained regarding claim 15, the '282 Patent states:

UBDs 130a-n may also modify the leaf image *in response to their respective compute node's access to its instance of the application environment*. For example, upon receiving a write request from their respective compute nodes for a sector X, the UBDs 130a-n will create an appropriate persistent mapping for sector X and then write sector X onto their respective second storage units 150a-n, where sector X can then be modified.

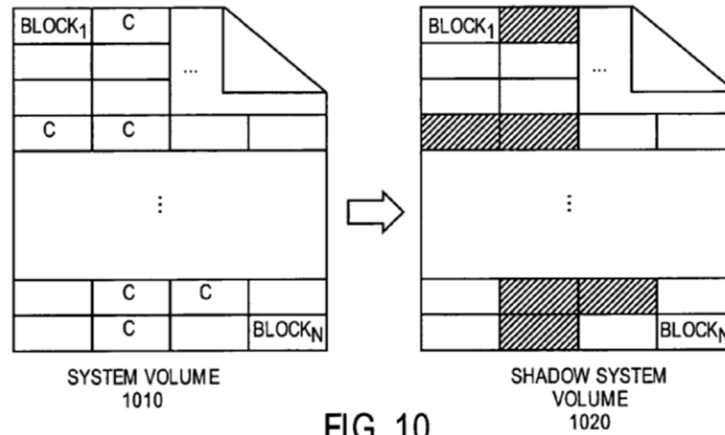
Ex.1001, 5:1-11 (emphasis added).

Another portion of the '282 Patent states similar subject matter:

Step 350 is modifying the leaf image in response to the compute node's access to its instance of the application environment. The modifying of the leaf image may include copying one or more data blocks from the root image to the leaf image, and modifying the data block in the leaf image.

Ex.1001, 6:23-27.

As also explained regarding claim 15, Birse discloses or suggests this subject matter. In Birse, “one or more shadow volumes [695], may be created for the NC client 150.” The “block device driver 154 has knowledge of the separate shadow volumes and system volumes 174 and redirects read and write requests to the appropriate volume.” Ex.1005, 7:35-53; *see also id.*, 11:30-32. Birse further explains that “the file system 152 generates a write request directed to the system volumes 160,” the “write request is received by the block device driver 154 and redirected by the block device driver 154 at step 420 by translating it into a write request directed to the user’s shadow volume on the NC server 170, and “the NC server 170 stores information associated with the write request in the shadow volume associated with the NC client 150.” *Id.*, 10:63-65, 11:6-17. As shown for example in FIG. 10 of Birse, blocks of the shadow volume (leaf image) that are modified in response to a write request (a sector X of the shadow volume) are persistently mapped (a “one-to-one block mapping in copy-on-write images”) and correspond to associated blocks of the original system volume (root image).



Id., 10:63-65, 11:6-17, 15:33-34.

Birse further explains that NC clients “boot by accessing a copy of the operating system over the network” and “[w]hen an NC client boots from the network and accesses the operating system from an NC server, the user's preferences are dynamically merged with the system environment provided to the NC client.” Ex.1005, 4:31-35, 9:2-5. Ex.1003, ¶¶332-338.

3. Claims 3 and 19

[3]/[19] “wherein the compute node comprises a thin-client workstation.”

Birse discloses that its NC clients can be “thin clients” that “generally refers to a disk-less computer system that relies upon external means, such as a server, to supply it with an operating system and application programs.” Ex.1005, 4:27-35. A POSITA would have understood that Birse’s disclosure of its NC clients as “thin clients” is equivalent to a thin-client workstation as described in the ’282 Patent. Ex.1003, ¶¶273-75, 339-40.

4. Claims 4 and 20

[4]/[20] “wherein the root image comprises an operating system.”

As explained in Section VII.A.1. regarding the independent claims, Birse specifically discloses that the system volumes comprise an operating system (e.g. operating system images 650, 660). Ex.1005, 12:44–47. Birse specifically discloses that “[a]n ‘operating system’ refers to a *program* that manages a computer’s internal functions and provides a means to control the computer’s operations and file system.” Ex.1005, 4:7-9. Ex.1003, ¶¶276-78, 341-42.

5. Claims 5 and 21

[5]/[21] “wherein the root image is concurrently accessible to a plurality of compute nodes.”

The Birse system can include a plurality of compute nodes (NC clients) as shown, for example, in FIG. 8 below.

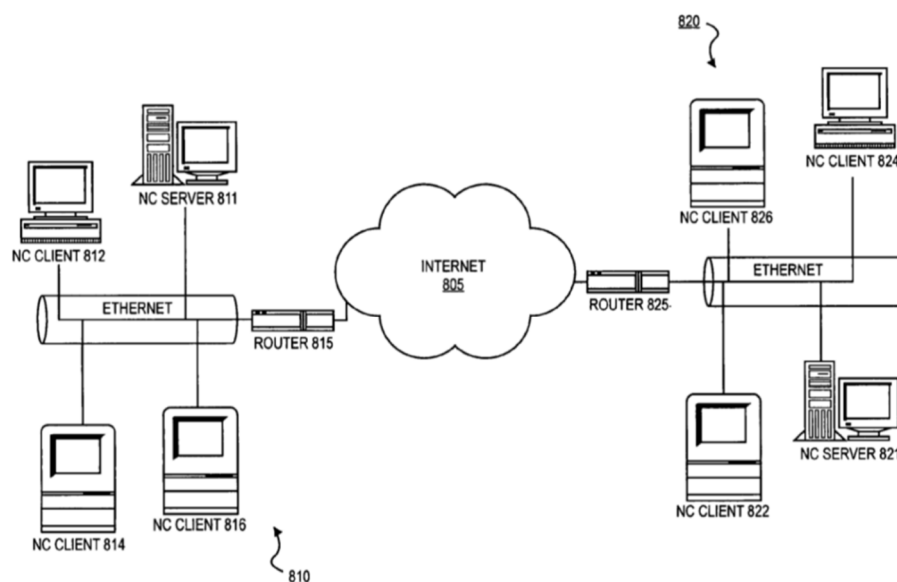


FIG. 8

The OS system volume (root image) is made concurrently accessible to the NC clients by the NC server “that services requests of NC clients” including “provid[ing] access to a copy of the operating system...in response to a boot request.” Ex.1005, 4:31-40. Thus, “when NC clients want to run an application, they access it from the NC server.” Ex.1005, 4:59-60. As explained regarding the independent claims, the system volume (root image) is provided in a location that is accessible to all NC clients (NC shared data folder 840), with NC client specific data (e.g., shadow volumes (leaf images)) being stored in NC client specific locations like folder 690, as shown in FIG. 6 below.

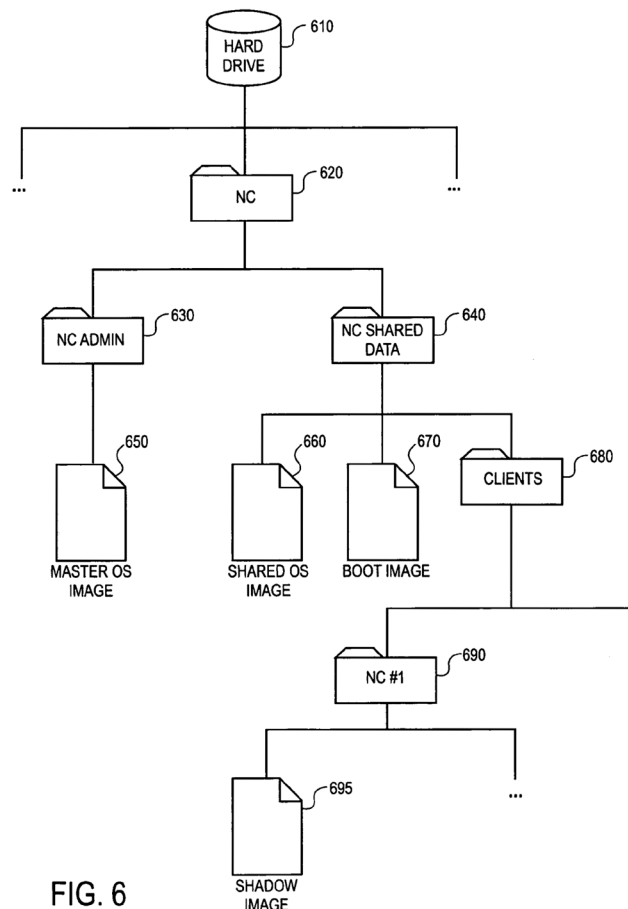


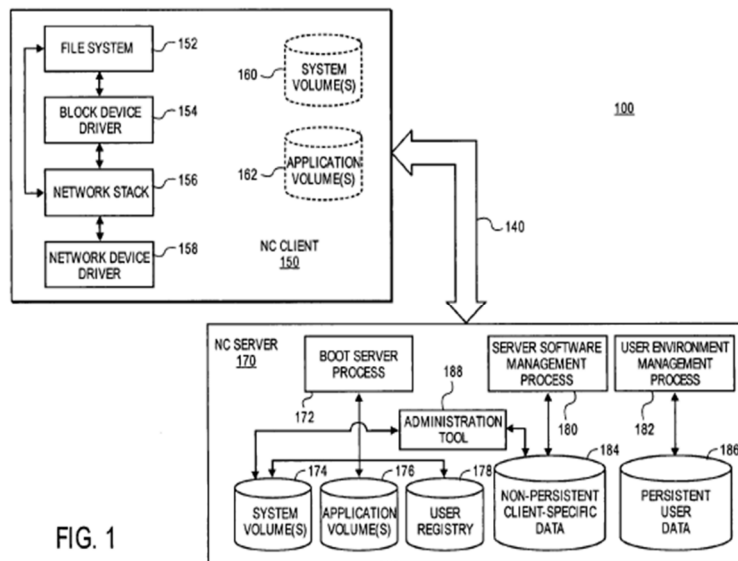
FIG. 6

Ex.1005, 12:30-52. Ex.1003, ¶¶279-82, 343-44.

6. Claims 6 and 22

[6]/[22] “wherein the first storage unit is remotely located from the compute node.”

As explained in Section VII.A.1. regarding the independent claims, Birse’s NC client 150 (compute node) is remote from the first storage unit (NC folders 630/640 of hard drive 610) because the first storage unit in Birse is on the NC server 170, which is remote from the NC client 150, as the NC server 170 and NC client 150 communicate over a data communication link 140, as shown in FIGS. 1 and 6 below.



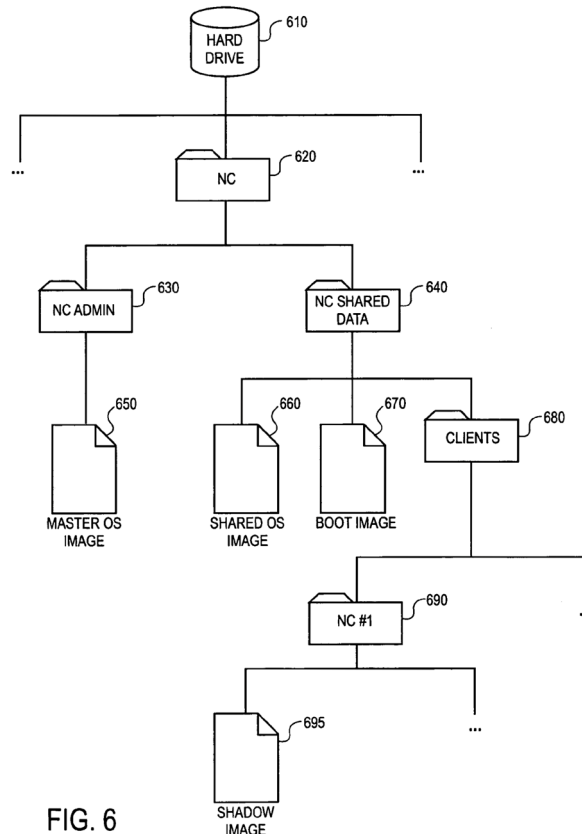


FIG. 6

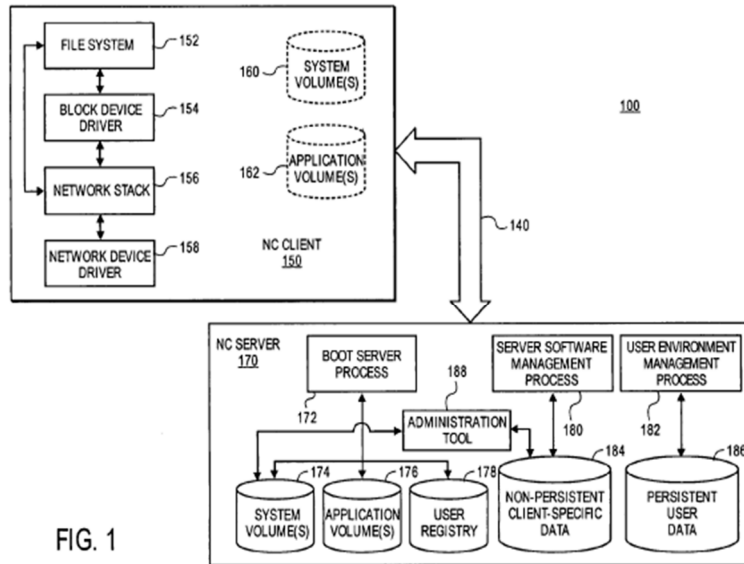
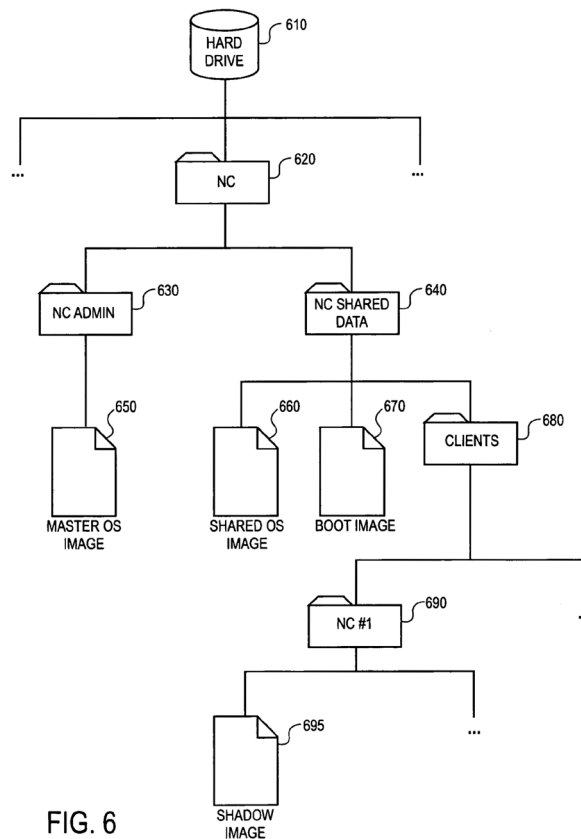
Birse further specifies that the storage on the NC server (e.g., hard drive 610) is a “remote storage medium on the NC server.” Ex.1005, 10:56-62. Ex.1003, ¶¶283-86, 345-46.

7. Claims 7 and 23

[7]/[23] “wherein the second storage unit is remotely located from the compute node.”

As explained in Section VII.A.1. regarding the independent claims, Birse’s NC client 150 (compute node) is remote from the second storage unit (NC folders 680/695 of hard drive 610) because the second storage unit in Birse is on the NC server 170, which is remote from the NC client 150, as the NC server 170 and NC

client 150 communicate over a communicate over data communication link 140, as shown in FIGS. 1 and 6 below.



Other portions of Birse emphasize the second storage unit is “remote” because Birse specifies that read/write requests to the shadow volumes on the second storage unit are redirected by the block device driver 154 “to a remote storage medium on the NC server.” Ex.1005, 10:56-62. Ex.1003, ¶¶287-90, 347-48.

8. Claims 8 and 24

[8]/[24] “wherein the second storage unit contains a block modification log for the compute node.”

As described in Section VII.A.1. with respect to claim elements [1.3] and [15.3], Birse discloses storing a leaf image (shadow volume/image 930, 695) for respective compute nodes (NC clients 150, 812, 814, 816, 822, 824, 826) on a second storage unit (NC client folders 690 #1...n on hard drive 610), where the leaf image includes new data blocks of the root image (write requests for new data) and changes to the blocks of the root image (write request for changes to operating system data): “In this example, the NC clients are numbered from 1 to N and the clients folder 680 includes a folder for each NC client. For example, an NC #1 folder 690 is an area for storing client-specific information, such as a shadow image 695, corresponding to NC client #1.” Ex.1005, 12:54–59.

Birse’s shadow volume “shadows the user system volume 924 by *storing modifications* that are made to the user system volume 924.” *Id.*, 14:49-53 (emphasis added). Therefore, the shadow volumes in Birse serve as block modification logs in

that they track blocks that have been modified as compared to the original system volumes. Ex.1003, ¶¶291-94, 349-50.

9. Claims 9 and 25

[9]/[25] “wherein the first storage unit [9:in [sic]] [25:is] contained within a first partition on a hard disk and the second storage unit is contained within a second partition on the hard disk.”

As an initial matter, the ’282 patent does not define or even use the term “partition” in its written description, only in the claims. Given that, and as described in Section VI above, a POSITA would have understood that the claimed partitions here can be any logical storage area on a hard disk: a folder or directory, a raw partition, an area of a database, etc. To the extent Patent Owner argues that the claimed partitions here must be raw partitions of a physical hard disk, a POSITA would have understood that a hard disk (*e.g.*, SCSI hard disk /dev/sda) can be easily partitioned using standard tools into several partitions (*e.g.*, /dev/sda1, /dev/sda2, /dev/sda3, and so on). Ex.1003, ¶¶295-96.

Birse discloses the first storage unit (holding the core system volumes and the user system volume), and the second storage units (holding the shadow system volumes) are contained on a same hard disk, but in different locations/partitions: “FIG. 6 is a simplified block diagram illustrating the layout of an exemplary

hierarchical directory structure that may be used by an NC server according to one embodiment of the present invention.” Ex.1005, 2:24–27.

“According to the embodiment depicted, a hard drive 610 of the NC server 170 or other mass storage device associated with the NC server 170 includes an NC folder (directory) 620.” Ex.1005, 12:30–37. Birse’s Figure 6 is reproduced below:

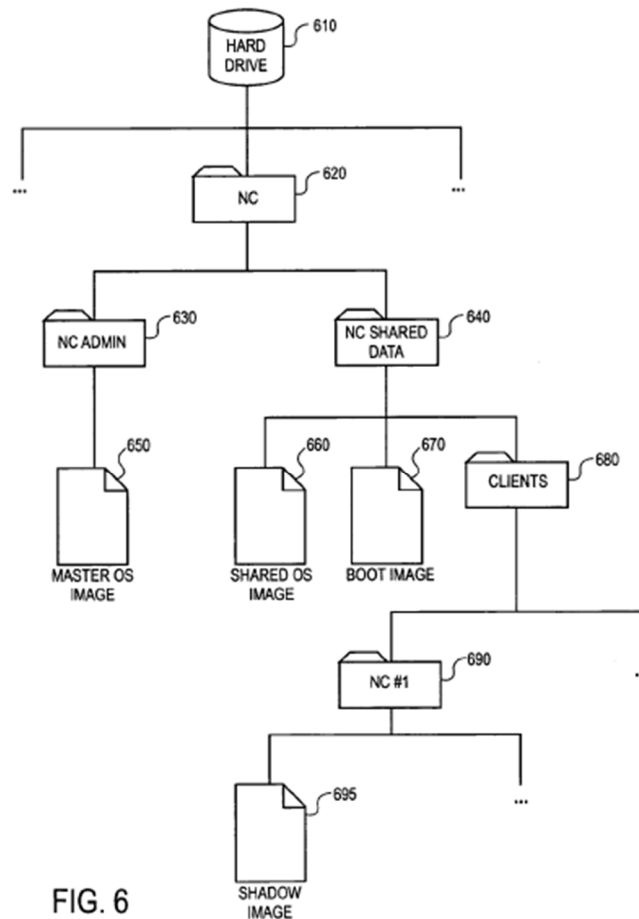


FIG. 6

Ex.1003, ¶¶297-99, 351-52.

10. Claim 10

[10] “wherein the union block device writes sector X on the second storage unit.”

Birse renders Claim 10 obvious for the same reasons as described in Section VII.A.1. with respect to claim element [15.7]. Ex.1003, ¶¶300-302, 328-31.

11. Claim 26

[26] “wherein merging occurs at an operational level between the first and second storage units and file system of the compute node.”

Birse renders Claim 26 obvious for the same reasons as described in Section VII.A.1. with respect to claim elements [1.4]-[1.6] pertaining to the union block device. Ex.1003, ¶¶251-63, 353-54.

12. Claims 13 and 29

[13]/[29] “wherein the first storage unit is read only.”

Birse further discloses the first storage unit being read-only: “The system volumes 174 include a protected, read-only, master copy of the operating system software.” Ex.1005, 5:13-14. Additionally, although Birse identifies that system volumes 660/924 accessed by the NC clients are “read-write” files (see FIG. 9 below), these system volumes are also effectively treated as “read only” in various embodiments of Birse.

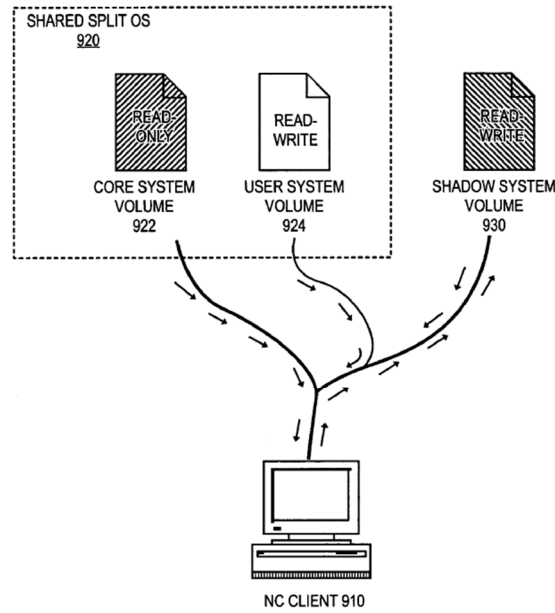


FIG. 9

As explained regarding the independent claims, in Birse, write requests to a system volume (i.e., “said first storage unit”) are redirected by the block device driver by translation into “a write request directed to the user’s shadow volume on the NC server.” Ex.1005, 11:8–11. Thus, because all write requests directed to a system volume are redirected to the user’s shadow volume, the system volume is “read-only” as required by claims 13 and 29. Ex.1003, ¶¶303-306, 355-56.

13. Claims 14 and 30

[14]/[30] “[14:further comprising][30:further comprising: creating] an intermediate image [30:on a third storage unit] between the root image and the leaf image, wherein the intermediate image comprises an application environment service group.”

Birse renders claims 14 and 30 obvious. Ex.1003, ¶¶307-317, 357-358. The '282 Patent describes that an intermediate image comprises versions of the root image from which a compute node's instance of the application environment is derived. Ex.1001, 4:58-61. The '282 Patent also describes that, as shown in FIG. 2. below, a second branch node 220 includes a service group 221 (service group 2.1) that “reflects a change to the application environment” and resides between the root image and the leaf images 150g-150i. Ex.1001, 5:41-52.

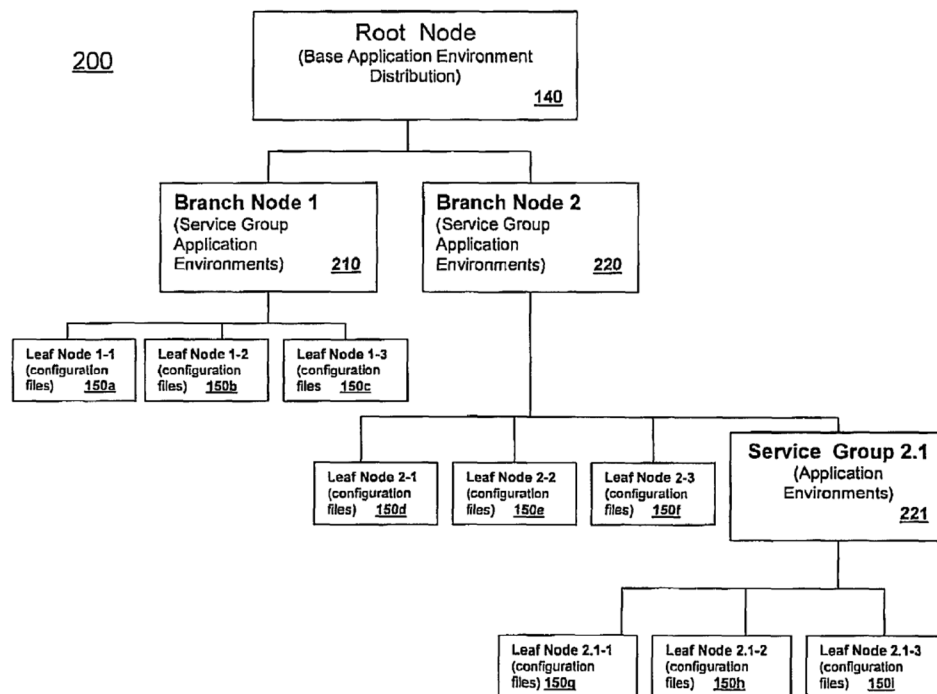


Figure 2

The '282 Patent explains that an application group (AG) “is a way to provide a group of compute nodes their own instantiation of an application environment” and that “[a]t creation time, an AG comprises the original root image.” Ex.1001,

5:19-29. Nodes shown in FIG. 2 can be “read-only” and others can be “read-write-modify.” Ex.1001, 5:32-33. Ex.1003, ¶¶307-310.

Likewise, as described herein with respect to independent claims 1 and 15, Birse discloses storing a master (read-only) OS image 650 and a shared OS image 660 (read-write), as shown in FIG. 6 of Birse below.

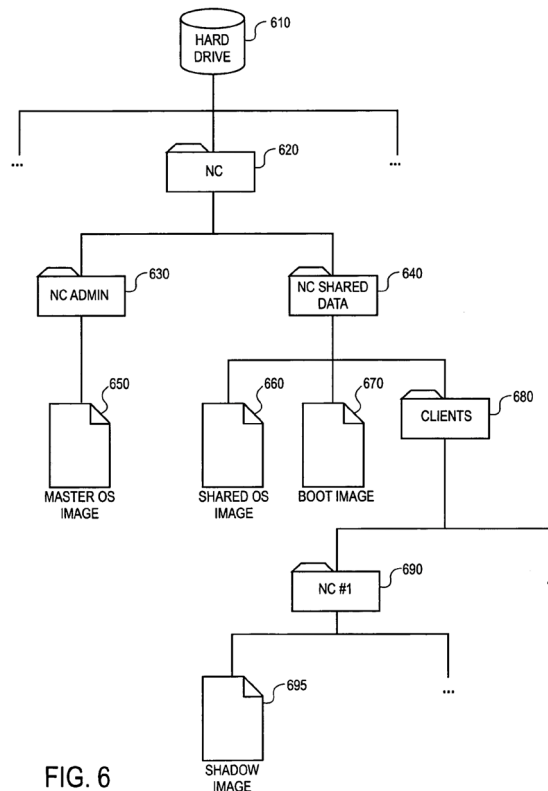


FIG. 6

In Birse, the shared OS image 660 can be stored in a third storage unit (NC shared data folder 640) and, while the “shared operating system image 660 is a read-write version of the read-only master operating system image 650” (Ex.1005, 12:44–47), changes can be made to the shared OS image 660 such that it acts as an intermediate image comprising an application environment service group.

Specifically, in Birse, an NC client 150 that “has appropriate access privileges” communicates with NC server 170 and NC server 170 copies one or more shared images, like the shared operating system image 650, to the NC client’s folder as a working copy. Ex.1005, 13:9-22. Changes can then be made to the working copy and a new shared images (e.g., a new shared OS image 660), can then be stored on the server 170 by the NC client committing the changes, which causes the new one or more shared images to become “available to any NC client that subsequently boots.” Ex.1005, 13:20-14:8. This process is illustrated in FIG. 7 of Birse below.

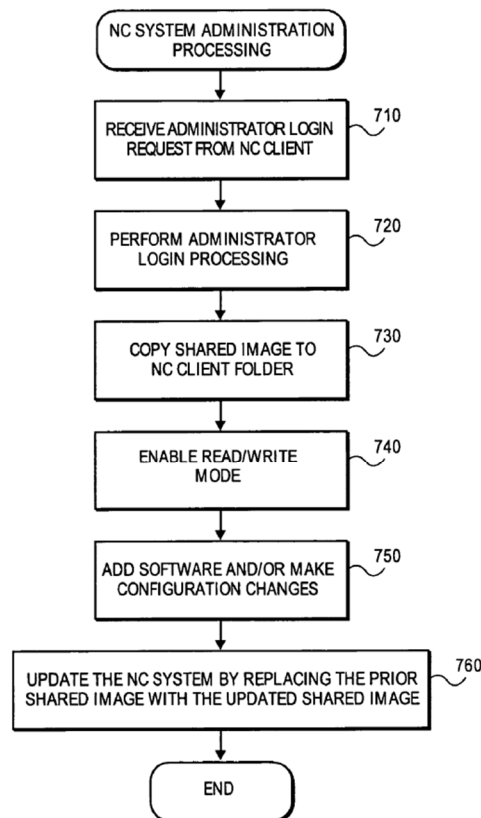


FIG. 7

Ex.1003, ¶¶311-313.

Birse further explains that this intermediate image can be made available to a certain group of NC clients or distributed and made available to many other networked connected NC clients. See FIG. 8 of Birse, below.

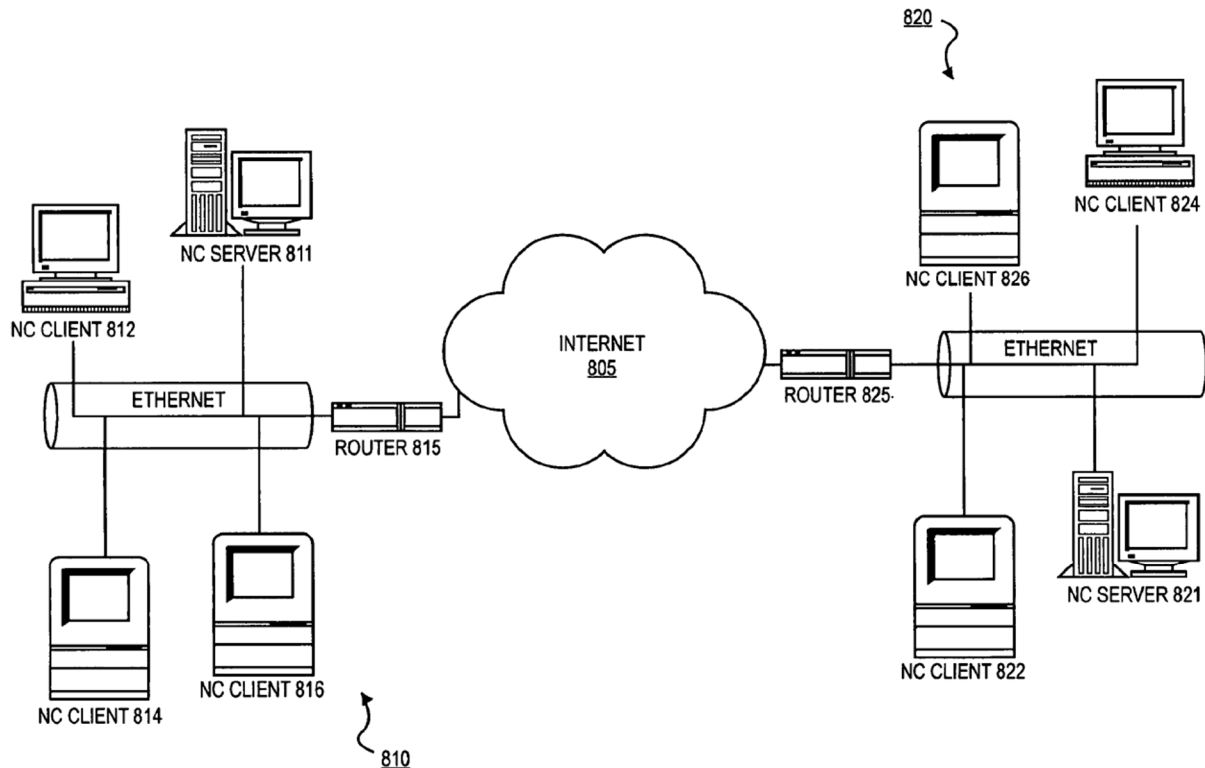


FIG. 8

Specifically, Birse describes, with reference to FIG. 8, that “two NC systems 810 and 820” can be “coupled via the Internet 805.” Birse contemplates “that the NC system configuration process described above with reference to FIG.7 may also be useful in connection with the reconfiguration of remote NC systems” and that “[f]or example, the administration tool 188 may be run from any of NC clients 812, 814, or 816 to install a new application program on local NC system 810 that is

desired by the users of the local NC system 810 as well as by the users of NC clients 822, 824, and 826 of remote NC system 820.” Then, “[a]fter completing the steps described above for installation on the local NC system 810, the administration tool 188 running on NC system 810 may replicate the modified shared images to the remote NC system 820. Ex.1005, 14:8-21. Ex.1003, ¶¶314-15.

Thus, as described and shown in FIGS. 6 and 7 above, in Birse, shared OS images 660 can be created that differ from the original master OS image (an intermediate image) on a third storage unit that is between the root image and the leaf image (as shown in FIG. 6 above where the changed shared OS image 660 is stored in folder 640 between the master OS image 650 and the shadow image(s) 695). This intermediate image in Birse comprises an application environment service group because, as explained above, in Birse, the intermediate image reflects a change from the original master OS image and is made available to NC clients so that this group of NC clients can use the revised application environment. Ex.1003, ¶¶316-317.

**VIII. GROUND 2: CLAIMS 2, 11, 12, 18, 27, AND 28 ARE OBVIOUS OVER
THE COMBINATION OF BIRSE AND BALLEW**

A. Motivation to Combine

Both Birse and Ballew are concerned with management of computing devices over a network and booting and/or initializing the computing devices with an operating system. For example, as explained in Sections VI-VII, Birse discloses computing devices that boot using a net-booted environment “by accessing a copy of the operating system over a network,” and in which one or more shadow volumes are created from the NC client 150 and stored with the non-persistent client-specific data 184 for the purpose of storing changes to the operating system made during execution by a particular computing device. Ex.1005, 4:31-35, 7:41-46.

As explained in Section VI, Ballew is concerned with an HPC system that uses HPC clusters of a server 102 with a grid 110 that “is a group of nodes 115 interconnected for increased processing power” for dynamically allocating processing of tasks. Ex.1006, ¶¶[0016]-[0017]. Like Birse, Ballew discusses that when an HPC server 102 “is first initialized or booted” “a unique execution environment” is provided “for each allocated node of virtual cluster 220,” where the execution environment can include an “operating system” and “a set of installed applications and data.” Ex.1006, ¶[0029].

A POSITA would have been motivated to modify Birse and/or Ballew such that the net-booted operating system processes disclosed in Birse could be used with

HPC servers and associated nodes like those disclosed in Ballew. At least one reason is that, while Ballew discloses providing an execution environment generally to nodes in an HPC system, Birse offers details on OS storage and distribution (*e.g.*, Birse’s master OS images and shadow images), such that a master OS image is stored by the server 102 in Ballew and the operating system of the execution environment for each node of the server 102 can be distributed using Birse’s shadow volumes.

Modifying Birse with Ballew’s HPC servers/cluster teachings would have been simple to a POSITA, and would not have required extensive experimentation or imposed undue burden – especially considering both contemplate distribution of application environments to computing devices. For the same reasons, the combination would have yielded predictable results (as HPC servers were well-understood to improve task processing power and allocation). A POSITA would also have a high expectation of success, as Ballew describes a need for distributing execution environments to nodes, and Birse provides a mechanism to do so. Ex.1003, ¶¶359-62.

B. Birse and Ballew meet all claim elements.

1. Claims 2 and 18

[2]/[18] “wherein the compute node comprises a server.”

Ballew discloses, as part of a server 102, a node that can “comprise any computer or processing device such as, for example, blades, general-purpose

personal computers (PC), Macintoshes, workstations, Unix-based computers, or any other suitable devices.” Ex.1006, ¶¶[0016]-[0017].

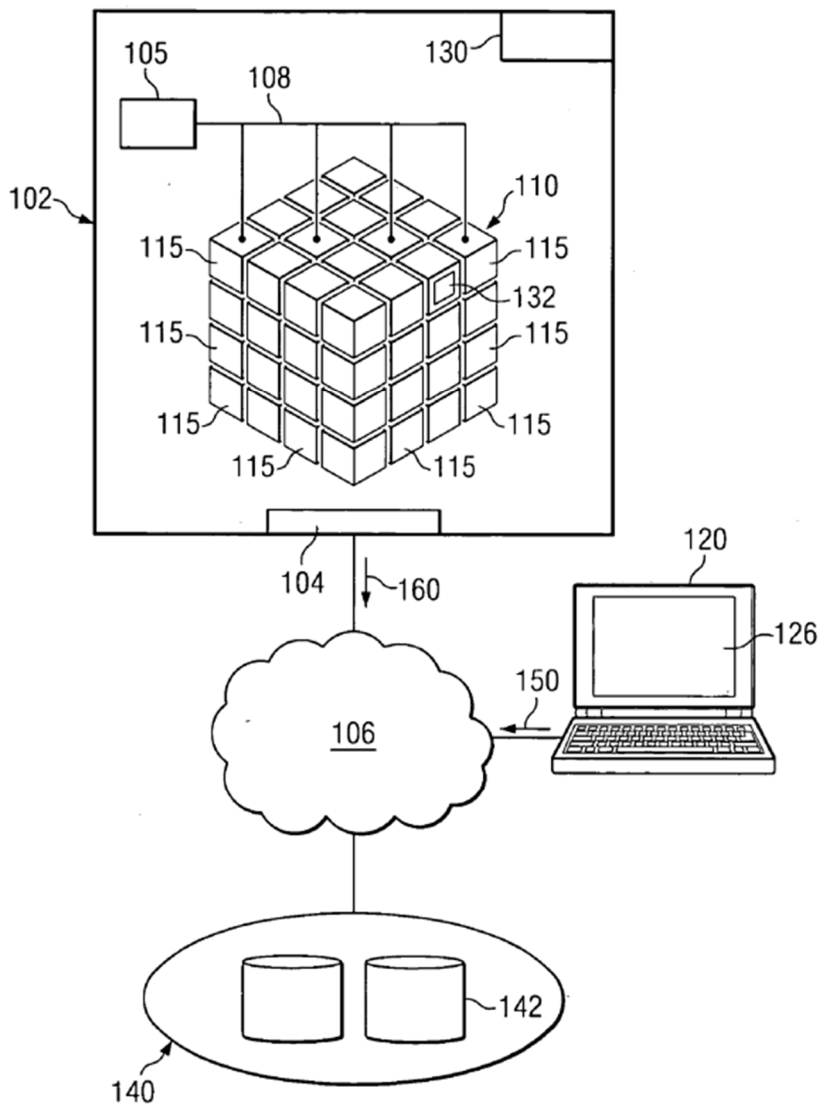
As explained in Section VII.A.1, Birse meets all elements of independent Claims 1 and 15. Further, as explained in the motivations to combine Birse and Ballew above, it would have been obvious to use Birse’s OS distribution processes with a server as described in Ballew, as Ballew discusses a need to distribute an operating system as part of an execution environment to Ballew’s nodes.

Therefore, Birse as combined with Ballew meets Claims 2 and 18. Ex.1003, ¶¶363-366, 379-80.

2. Claims 11 and 27

[11]/[27] “wherein the [11:system][27:method] operates in a high performance computing cluster.”

As explained in Section VII.A.1., Birse meets all elements of independent Claims 1 and 15. Ballew further discloses a “high performance computer system and method” in which a “high performance computing (HPC) system 100” is used for “executing software, applications, and processes.” Ex.1006, Title, ¶[0014]. See FIG. 1 of Ballew, below.



In Ballew, when the HPC server 102 “is first initialized or booted” a “cluster management engine 130 may dynamically allocate various portions of grid 110 to one or more virtual clusters 220 based on, for example, predetermined policies,” and can “also manage and provide a unique execution environment for each allocated node of virtual cluster 220.” Ex.1006, ¶[0029]. Ballew explains that “[t]he execution

environment may consist of the hostname, IP address, operating system, configured services, local and shared file systems, and a set of installed applications and data.”

Id.

Further, as explained in the motivations to combine Birse and Ballew above, high performance computing clusters were well-known prior to the '282 Patent, and it would be obvious to operate systems and methods that deal with distributing application environments, like that of Birse, in a high performance computing cluster. Ex. 1003, ¶¶367-71.

The Background of the '282 Patent itself indicates that HPC clusters were known prior to the '282 Patent, explaining that “[o]ver the years, as the interne [sic] has expanded and computers have multiplied, the need for clustered computing such as High Performance Computing (HPC) has increased” and that “[c]lustered computing involves multiple compute nodes, usually a server grid, that work together to achieve a common task.” Ex. 1001, 1:20-25. The '282 Patent further asserts that there were issues with “low cluster bring-up time” because complete clones of master OS images would have to be created for each node in the cluster and those complete clones would have to be updated to update the OS of the cluster. *Id.*, 1:39-57.

Birse as combined with Ballew provides a solution to this same problem, such that one master OS image can be stored and the operating system of the execution

environment for each node of Ballew's server 102 can be distributed using Birse's shadow volumes.

Therefore, Birse as combined with Ballew meets Claims 11 and 27. Ex. 1003, ¶¶371-73, 381-82.

3. Claims 12 and 28

[12]/[28] “wherein the system operates in a grid computing cluster.”

As explained above regarding Claims 11 and 27, the Background of the '282 Patent admits that clustered computing “involves multiple compute nodes, usually a server grid.” Ex. 1001, 1:20-25. The clustered computing described in Ballew uses a grid 110 comprising the plurality of nodes 115, where the grid 110 “is a group of nodes 115 interconnected for increased processing power.” Ex.1006, ¶¶[0016]-[0017]. FIGS. 2C and 2D, shown below, illustrate examples in which a grid 210 has virtual clusters 220 allocated within the grid.

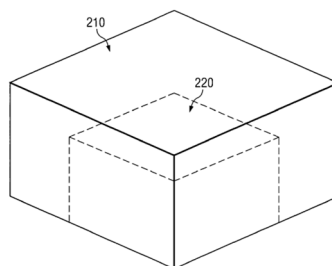


FIG. 2C

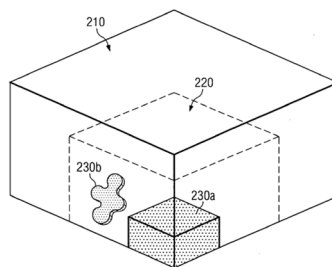


FIG. 2D

Ballew indicates that multiple servers 102, e.g., a “server pool,” each having a grid can also be used. Ex.1006, ¶¶[0016]-[0017].

Therefore, Birse as combined with Ballew meets Claims 12 and 28. Ex. 1003, ¶¶374-78, 383-84.

IX. GROUND 3: CLAIMS 8 AND 24 ARE OBVIOUS OVER THE COMBINATION OF BIRSE AND WU

A. Motivation to Combine

Both Birse and Wu are concerned with “copy-on-write” schemes in which reads or writes directed to a main volume are redirected to a virtual volume. Ex. 1005, 14:56-61 (“copy-on-write” redirecting to a shadow volume); Ex.1007, 4:61-5:4 (“copy-on-write” redirecting to a snapshot). For example, as explained in Sections VI-VII, Birse discloses creating “one or more shadow volumes” “for the purpose of storing changes to the operating system [stored in a core system volume] made during execution by a particular computing device.” Ex.1005, 7:41-46.

As explained in Section VI, Wu discloses using a modification log to track modifications to the original volume between any two snapshots, such that, “[e]ach time a modification is detected, modification manager 212 may log the time at which the modification occurred, the address(es) of the modified block(s), and the pre- and post-modification values of the modified block(s).” Ex.1007, 6:14-23. In the event a snapshot is deleted, the modifications stored in the modification log can be used to reconstruct the snapshot using another snapshot. Ex.1007, 7:26-29. Wu explains that

storing snapshot volumes can be storage intensive, and thus proposes using its smaller footprint modification logs to track block changes. Ex.1007, 1:66-2:42.

A POSITA would have been motivated to modify Birse based on Wu's disclosure to include Wu's modification log in Birse's system such that block modifications to Birse's system volumes are tracked in addition to, or in place of, Birse's shadow volumes. Doing so would have provided the Birse system with another way to track modifications to system volumes with a more efficient disk space usage. Furthermore, it would have additionally allowed for deleted shadow volumes to be restored if needed using Wu's modification log.

Modifying Birse to use Wu's modification log teachings would have been simple to a POSITA and would not have required extensive experimentation or imposed undue burden – especially considering that both contemplate management of copy-on-write volumes. For the same reasons, the combination would have yielded predictable results, and a POSITA would also have a high expectation of success, as both Birse and Wu describe a need for limiting disk space of shadow volumes and snapshots and Wu's modification log provides a detailed solution to this problem while also providing a mechanism for restoring deleted data. Ex. 1005, 15:15-42; Ex.1007, 1:66-2:42. Ex.1003, ¶¶385-88.

B. Birse and Wu meet all claim elements.

1. Claims 8 and 24

[8]/[24] “wherein the second storage unit contains a block modification log for the compute node.”

To the extent Birse does not explicitly disclose a block modification log, Wu discloses a block modification log that is used to log and track modifications to an original volume between any two snapshots. Ex.1007, 6:14-20. As shown in FIG. 2 of Wu, below, the block modification log includes “the time at which the modification occurred, the address(es) of the modified block(s), and the pre- and post-modification values of the modified block(s).”

Modification Log 260

	Timestamp	Block Address	Pre-Modification Value	Post-Modification Value
Modification 1	Between T1 & T2	Address 1	Value A	Value B
Modification 2	Between T1 & T2	Address 2	Value X	Value Y
Modification 3	Between T2 & T3	Address 3	Value C	Value D

FIG. 5

Birse combined with Wu, as described above, would provide for a modification log to be used in Birse’s system to track changes to system volume blocks. Therefore, Birse as combined with Wu meets Claims 8 and 24. Ex. 1003, ¶¶389-91, 392-93.

X. GROUND 4: CLAIMS 9, 17, AND 25 ARE OBVIOUS OVER THE COMBINATION OF BIRSE AND KLEINSCHNITZ

A. Motivation to Combine

Both Birse and Kleinschnitz are concerned with storing changes to system volumes for use in virtual environments. Ex. 1005, 14:26-15:7 (using shadow volumes to store changes to system volume); Ex.1008, [0003], [0007], [0016], [0044] (using snapshots to store incremental changes to a compute state). For example, as explained in Sections VI-VII, Birse discloses computing devices that boot using a net-booted environment “by accessing a copy of the operating system over a network,” and in which one or more shadow volumes are created from the NC client 150 and stored with the non-persistent client-specific data 184 for the purpose of storing changes to the operating system made during execution by a particular computing device. Ex.1005, 4:31-35, 7:41-46.

As explained in Section VI, Kleinschnitz discloses using snapshots that include “incremental images” reflecting “the state of the computer’s storage” and storing the snapshots at another storage location. Ex.1008, [0003], [0035], [0059]. Kleinschnitz explains that “an incremental image 104 may be merged with a prior complete snapshot image 104 to form a complete and more recent snapshot image.” Ex.1008, [0071]. Kleinschnitz also discloses that storage disks can be partitioned, such as “by placing user data in one partition and system data and software in another partition.” Ex.1008, [0079].

A POSITA would have been motivated to modify Birse based on Kleinschnitz's disclosure to include Kleinschnitz's merging/reconciliation of incremental images in Birse's system such that the one or more shadow volumes are merged to create an updated complete volume. A POSITA would have understood that this "reconciling" is nothing more than merging snapshots, a process that naturally happens when snapshots are removed or "collapsed" together to make room. A POSITA would have also been motivated to modify Birse based on Kleinschnitz's disclosure to store system volumes (650,660,922,924) and shadow volumes (695,930) in different partitions so that, like in Kleinschnitz, user data (shadow volumes) are stored in one partition and system data (system volumes) are stored in another partition.

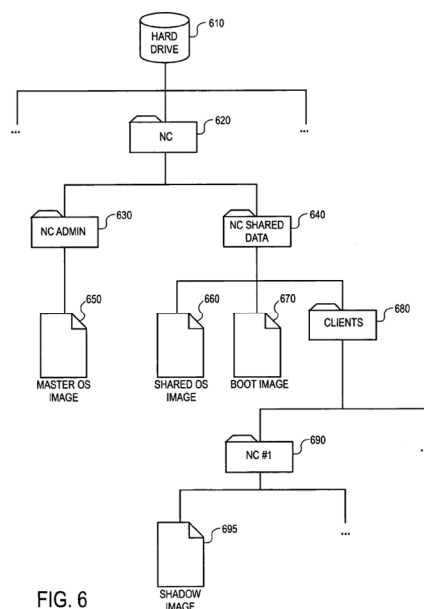
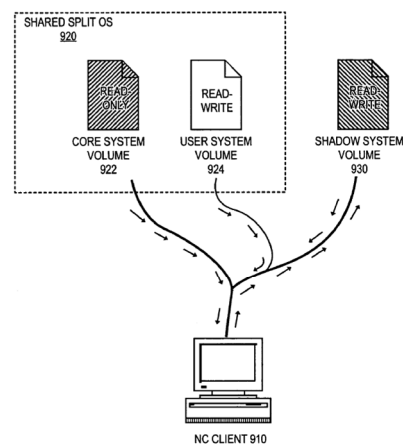
Modifying Birse to use Kleinschnitz's merging/reconciliation and partition would have been simple to a POSITA, and would not have required extensive experimentation or imposed undue burden – especially considering both contemplate management and storage of system volume changes. For the same reasons, the combination would have yielded predictable results and a POSITA would also have a high expectation of success, as both Birse and Kleinschnitz describe snapshot reconciliation and disk partitioning techniques that were both already well-known before the '282 Patent. Ex.1003, ¶¶394-99.

B. Birse and Kleinschnitz meet all claim elements.

1. Claims 9 and 25

[9]/[25] “wherein the first storage unit [9:in [sic]] [25:is] contained within a first partition on a hard disk and the second storage unit is contained within a second partition on the hard disk.”

Kleinschnitz discloses that storage disks can be partitioned, such as “by placing user data in one partition and system data and software in another partition.” Ex.1008, [0079]. As explained above, it would have been obvious to modify Birse based on Kleinschnitz’s disclosure to store Birse’s system volumes (650,660,922,924) and shadow volumes (695,930), shown, e.g., in FIGS. 6 and 9 of Birse below, in different partitions such that, like in Kleinschnitz, user data (shadow volumes) are stored in one partition and system data (system volumes) are stored in another partition.



Partitioning the data in this way would have served to create a clear distinction between storage/access locations for system volumes and shadow volumes.

Therefore, Birse as combined with Kleinschnitz meets Claims 9 and 25. Ex. 1003, ¶¶400-403, 409-10.

2. Claim 17

[17] “reconciling the root image and the leaf image to form a new root image.”

Kleinschnitz discloses using snapshots that include “incremental images” reflecting “the state of the computer’s storage” and storing the snapshots at another storage location. Ex.1008, [0003], [0035], [0059]. Kleinschnitz explains that “an incremental image 104 may be merged with a prior complete snapshot image 104 to form a complete and more recent snapshot image.” Ex.1008, [0071].

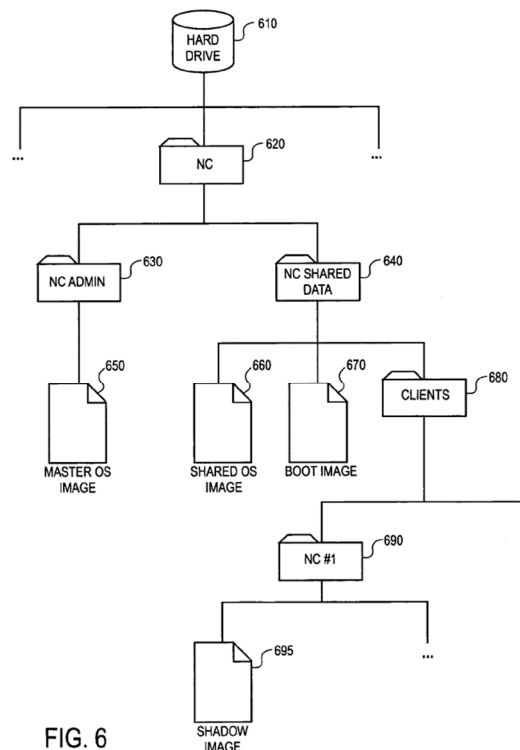


FIG. 6

As explained above, a POSITA would have been motivated to modify Birse based on Kleinschnitz's disclosure to include Kleinschnitz's merging/reconciliation of incremental images in Birse's system such that the one or more shadow volumes are merged to create an updated complete volume. A POSITA would have understood that this "reconciling" is useful to make room in storage for new shadow images. The '282 Patent provides the same reasons for reconciling its leaf images with the root images, stating such reconciliation "may be desirable if, for example, the leaf image has grown to exceed a particular disk quota." Ex.1001, 6:30-32.

Therefore, Birse as combined with Kleinschnitz meets Claim 17. Ex. 1003, ¶¶404-408.

XI. CONCLUSION

Claims 1-30 are unpatentable under Pre-AIA §103. Petitioners request cancellation of claims 1-30.

Dated: October 10, 2025

Respectfully submitted,

By: /Keith D. Harden/
Keith D. Harden
Registration No. 74,472

CERTIFICATE OF WORD COUNT UNDER 37 C.F.R. § 42.24(d)

The undersigned certifies that the word count of the foregoing PETITION FOR *INTER PARTES* REVIEW, excluding the table of contents, table of authorities, exhibit list, mandatory notices, certificate of service, certificate of word count, and any appendix of exhibits or claim listing, is 13,915.

Dated: October 10, 2025

/Keith D. Harden/
Keith D. Harden
Registration No. 74,472

CERTIFICATE OF SERVICE

The undersigned certifies that a copy of the foregoing PETITION FOR *INTER PARTES* REVIEW and all exhibits identified herein were served on October 10, 2025 at the following correspondence address of record for the subject patent:

Anthony Murabito
MURABITO HAO & BARNES LLP
1500 E Hamilton Ave, Suite 118
Campbell, CA 95008

via FedEx Overnight

Courtesy Copy via electronic mail to Patent Owner's Known Counsel:

Brandon R. Theiss
Daniel H. Golub
Ryan O'Donnell
Dawn C. Kerner
Volpe Koenig
30 South 17th Street, 18th Floor
Philadelphia, PA 19103
btheiss@vklaw.com
dgolub@vklaw.com
rodonnell@vklaw.com
dkerner@vklaw.com

Jonathan K. Waldrop
Kasowitz Benson Torres LLP
333 Twin Dolphin Drive, Suite 200
Redwood Shores, CA 94065
jwaldrop@kasowitz.com
IntellectualVentures@kasowitz.com

Dated: October 10, 2025

/Keith D. Harden/
Keith D. Harden
Registration No. 74,472