

UNITED STATES PATENT AND TRADEMARK OFFICE

---

BEFORE THE PATENT TRIAL AND APPEAL BOARD

---

DOCKER INC.,  
Petitioner,  
v.

INTELLECTUAL VENTURES II LLC,  
Patent Owner.

---

Case No. IPR2025-00840

U.S. Patent No. 8,332,844

---

**Petition for *Inter Partes* Review of U.S. Patent No. 8,332,844**

Mail Stop **PATENT BOARD**  
Patent Trial and Appeal Board  
U.S. Patent and Trademark Office  
P.O. Box 1450  
Alexandria, VA 22313-1450

**TABLE OF CONTENTS**

|                                                                                            | <u>Page</u> |
|--------------------------------------------------------------------------------------------|-------------|
| PETITIONER’S EXHIBIT LIST .....                                                            | vi          |
| TABLE OF ABBREVIATIONS .....                                                               | vii         |
| MANDATORY NOTICES.....                                                                     | viii        |
| A.    Real Parties-in-Interest [37 C.F.R. § 42.8(b)(1)] .....                              | viii        |
| B.    Related Matters [37 C.F.R. § 42.8(b)(2)].....                                        | viii        |
| C.    Lead and Back-Up Counsel [37 C.F.R. §42.8(b)(3)].....                                | ix          |
| D.    Service Information [37 C.F.R. § 42.8(b)(4)] .....                                   | ix          |
| I.    INTRODUCTION .....                                                                   | 1           |
| II.   REQUIREMENTS OF <i>INTER PARTES</i> REVIEW .....                                     | 3           |
| A.    Standing.....                                                                        | 3           |
| B.    Identification of Challenge and Relief Requested .....                               | 3           |
| 1.    U.S. Patent No. 6,618,736 (“Menage”) (EX1004) .....                                  | 4           |
| 2.    U.S. Patent No. 7,395,324 (“Murphy”) (EX1005) .....                                  | 4           |
| 3.    U.S. Patent No. 7,089,300 (“Birse”) (EX1006).....                                    | 4           |
| 4.    U.S. Patent No. 7,398,382 (“Rothman”) (EX1007) .....                                 | 5           |
| C.    How the Challenged Claims Are To Be Construed Under 37<br>C.F.R. §42.104(b)(3) ..... | 5           |
| D.    How the Challenged Claims Are Unpatentable Under 37 C.F.R.<br>§42.104(b)(4).....     | 5           |
| E.    Supporting Evidence Under 37 C.F.R. §42.104(b)(5) .....                              | 6           |
| F.    Payment of Fees .....                                                                | 6           |
| III.  TECHNICAL BACKGROUND .....                                                           | 6           |

|     |                                                                            |    |
|-----|----------------------------------------------------------------------------|----|
| A.  | The '844 Patent and Summary of the Alleged Invention.....                  | 6  |
| B.  | Prosecution History .....                                                  | 8  |
| C.  | Person of Ordinary Skill in the Art .....                                  | 9  |
| IV. | GROUND OF UNPATENTABILITY.....                                             | 9  |
| A.  | Ground 1: Obviousness over Menage (Claims 1–13).....                       | 9  |
| 1.  | Summary of Menage.....                                                     | 9  |
| 2.  | Detailed Analysis of Ground 1 .....                                        | 17 |
| a.  | Independent Claim 1 .....                                                  | 17 |
| b.  | Claim 2.....                                                               | 28 |
| c.  | Claim 3.....                                                               | 29 |
| d.  | Claim 4.....                                                               | 30 |
| e.  | Claim 5.....                                                               | 32 |
| f.  | Claim 6.....                                                               | 33 |
| g.  | Independent Claim 7.....                                                   | 33 |
| h.  | Claim 8.....                                                               | 35 |
| i.  | Claim 9.....                                                               | 35 |
| j.  | Claim 10.....                                                              | 36 |
| k.  | Claim 11.....                                                              | 36 |
| l.  | Claim 12.....                                                              | 37 |
| m.  | Claim 13.....                                                              | 37 |
| B.  | Ground 2: Obviousness over Menage in view of Murphy<br>(Claims 14–27)..... | 37 |
| 1.  | Summary of Murphy.....                                                     | 37 |

|    |                                                                                     |    |
|----|-------------------------------------------------------------------------------------|----|
| 2. | Motivation to Combine Menage and Murphy and Reasonable Expectation of Success ..... | 40 |
| 3. | Detailed Analysis of Ground 2 .....                                                 | 42 |
| a. | Independent Claim 14.....                                                           | 42 |
| b. | Claim 15.....                                                                       | 47 |
| c. | Claim 16.....                                                                       | 47 |
| d. | Claim 17.....                                                                       | 48 |
| e. | Claim 18.....                                                                       | 48 |
| f. | Independent Claim 19.....                                                           | 48 |
| g. | Claim 20.....                                                                       | 50 |
| h. | Claim 21.....                                                                       | 51 |
| i. | Claim 22.....                                                                       | 51 |
| j. | Independent Claim 23.....                                                           | 51 |
| k. | Claim 24.....                                                                       | 54 |
| l. | Claim 25.....                                                                       | 54 |
| m. | Claim 26.....                                                                       | 55 |
| n. | Claim 27.....                                                                       | 56 |
| C. | Ground 3: Obviousness over Birse in view of Rothman (Claims 1–13).....              | 56 |
| 1. | Summary of Birse .....                                                              | 56 |
| 2. | Summary of Rothman .....                                                            | 56 |
| 3. | Motivation to Combine Birse and Rothman and Reasonable Expectation of Success ..... | 57 |
| 4. | Detailed Analysis of Ground 3 .....                                                 | 58 |

|    |                                                                                      |    |
|----|--------------------------------------------------------------------------------------|----|
| a. | Independent Claim 1.....                                                             | 58 |
| b. | Claim 2.....                                                                         | 69 |
| c. | Claim 3.....                                                                         | 69 |
| d. | Claim 4.....                                                                         | 69 |
| e. | Claim 5.....                                                                         | 71 |
| f. | Claim 6.....                                                                         | 72 |
| g. | Independent Claim 7.....                                                             | 72 |
| h. | Claim 8.....                                                                         | 73 |
| i. | Claim 9.....                                                                         | 74 |
| j. | Claim 10.....                                                                        | 74 |
| k. | Claim 11.....                                                                        | 75 |
| l. | Claim 12.....                                                                        | 75 |
| m. | Claim 13.....                                                                        | 75 |
| D. | Ground 4: Obviousness over Birse in view of Murphy<br>(Claims 14–27).....            | 75 |
| 1. | Motivation to Combine Birse and Murphy and<br>Reasonable Expectation of Success..... | 75 |
| 2. | Detailed Analysis of Ground 4 .....                                                  | 76 |
| a. | Independent Claim 14.....                                                            | 76 |
| b. | Claim 15.....                                                                        | 80 |
| c. | Claim 16.....                                                                        | 80 |
| d. | Claim 17.....                                                                        | 80 |
| e. | Claim 18.....                                                                        | 80 |
| f. | Independent Claim 19.....                                                            | 80 |

|     |                                          |    |
|-----|------------------------------------------|----|
| g.  | Claim 20.....                            | 81 |
| h.  | Claim 21.....                            | 82 |
| i.  | Claim 22.....                            | 82 |
| j.  | Independent Claim 23.....                | 82 |
| k.  | Claim 24.....                            | 83 |
| l.  | Claim 25.....                            | 83 |
| m.  | Claim 26.....                            | 84 |
| n.  | Claim 27.....                            | 84 |
| V.  | SECONDARY CONSIDERATIONS .....           | 85 |
| VI. | CONCLUSION.....                          | 85 |
|     | APPENDIX: CHALLENGED CLAIM LISTING ..... | 88 |

**PETITIONER'S EXHIBIT LIST**

| <b>Exhibit No.</b> | <b>Description</b>                                                                                                                                                                                  |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1001               | U.S. Patent No. 8,332,844                                                                                                                                                                           |
| 1002               | File History for U.S. Patent No. 8,332,844                                                                                                                                                          |
| 1003               | Declaration of Erez Zadok, Ph.D.                                                                                                                                                                    |
| 1004               | U.S. Patent No. 6,618,736                                                                                                                                                                           |
| 1005               | U.S. Patent No. 7,395,324                                                                                                                                                                           |
| 1006               | U.S. Patent No. 7,089,300                                                                                                                                                                           |
| 1007               | U.S. Patent No. 7,398,382                                                                                                                                                                           |
| 1008               | File History for U.S. Patent Appl. No. 11/395,816                                                                                                                                                   |
| 1009               | A. Silberschatz & P.B. Galvin, <i>Operating System Concepts</i> (4th ed. 1994)                                                                                                                      |
| 1010               | D.P. Bovet & M. Cesati, <i>Understanding the Linux Kernel</i> (1st ed. 2000)                                                                                                                        |
| 1011               | U.S. Patent No. 5,313,646                                                                                                                                                                           |
| 1012               | File History for U.S. Patent Appl. No. 11/026,622                                                                                                                                                   |
| 1013               | Excerpts from Microsoft Computer Dictionary (3rd ed. 1997)                                                                                                                                          |
| 1014               | Internet Small Computer Systems Interface (iSCSI), RFC 3720, IETF (April 2004), <a href="https://www.rfc-editor.org/rfc/pdf/rfc3720.txt.pdf">https://www.rfc-editor.org/rfc/pdf/rfc3720.txt.pdf</a> |
| 1015               | Dave Hitz <i>et al.</i> , Network Appliance Inc., <i>File System Design for an NFS File Server Appliance</i> , USENIX 1994                                                                          |
| 1016               | U.S. Patent No. 7,334,095                                                                                                                                                                           |
| 1017               | Hugo Patterson <i>et al.</i> , Network Appliance Inc., <i>File System Based Asynchronous Mirroring for Disaster Recovery</i> , USENIX 2002                                                          |
| 1018               | U.S. Patent No. 6,668,264                                                                                                                                                                           |

## TABLE OF ABBREVIATIONS

| Abbreviation      | Term                                     |
|-------------------|------------------------------------------|
| '477 application  | U.S. Patent Appl. No. 11/709,477         |
| '816 application  | U.S. Patent Appl. No. 11/395,816         |
| '622 application  | U.S. Patent Appl. No. 11/026,622         |
| '844 patent       | U.S. Patent No. 8,332,844 (EX1001)       |
| AIA               | America Invents Act                      |
| Birse             | U.S. Patent No. 7,089,300 (EX1006)       |
| Challenged Claims | Claims 1–27 of U.S. Patent No. 8,332,844 |
| Fig.              | Figure                                   |
| IPR               | <i>inter partes</i> review               |
| Menage            | U.S. Patent No. 6,618,736 (EX1004)       |
| Murphy            | U.S. Patent No. 7,395,324 (EX1005)       |
| Patent Owner      | Intellectual Ventures II LLC             |
| Petitioner        | Docker Inc.                              |
| POSITA            | person[s] of ordinary skill in the art   |
| Rothman           | U.S. Patent No. 7,398,382 (EX1007)       |

**MANDATORY NOTICES**

**A. Real Parties-in-Interest [37 C.F.R. § 42.8(b)(1)]**

The real party-in-interest is Petitioner Docker Inc. No other entity is funding, controlling, or directing this Petition, or otherwise has had an opportunity to control or direct this Petition or Petitioner's participation in any resulting *inter partes* review (IPR).

**B. Related Matters [37 C.F.R. § 42.8(b)(2)]**

The '844 patent is presently the subject of litigation in an action for declaratory judgment brought by Assurant, Inc., seeking a declaration of noninfringement of the '844 patent. *Assurant, Inc. v. Intellectual Ventures I LLC*, No. 1:24-cv-00344 (D. Del. filed Mar. 15, 2024). The '844 patent is currently asserted in *Intellectual Ventures I LLC v. Southwest Airlines Co.*, No. 7:24-cv-00277 (W.D. Tex. filed Nov. 2, 2024); *Intellectual Ventures I LLC v. Am. Airlines, Inc.*, 4:24-cv-00980 (E.D. Tex. filed Nov. 2, 2024); *Intellectual Ventures I LLC v. The Bank of N.Y. Mellon Corp.*, No. 3:25-cv-00631 (N.D. Tex. filed Mar. 15, 2025); and *Intellectual Ventures I LLC v. Nationwide Mut. Ins. Co.*, 3:25-cv-00632 (N.D. Tex. filed Mar. 15, 2025).

Additionally, the '844 patent is the subject of a petition for *inter partes* review filed by American Airlines, Inc. and Southwest Airlines Co. *Am. Airlines, Inc. et al. v. Intellectual Ventures II LLC*, IPR2025-00931 (filed Apr. 30, 2025).

**C. Lead and Back-Up Counsel [37 C.F.R. §42.8(b)(3)]**

Pursuant to 37 C.F.R. §§42.8(b)(3), 42.8(b)(4), and 42.10(a), Petitioner provides the following designation of counsel:

| <b>Lead Counsel for Petitioner</b>                                                                                                                                                                                                        | <b>Back-Up Counsel for Petitioner</b>                                                                                                                                                                                                           |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Brian M. Buroker (Reg. No. 39,125)<br>Gibson, Dunn & Crutcher LLP<br>1700 M St. NW<br>Washington, DC 20036<br>Phone: (202) 955-8500<br>Fax: (202) 467-0539<br>Email: <a href="mailto:bburoker@gibsondunn.com">bburoker@gibsondunn.com</a> | Nathan Curtis (Reg. No. 70,471)<br>Gibson, Dunn & Crutcher LLP<br>2001 Ross Avenue, Suite 2100<br>Dallas, TX 75201<br>Phone: (214) 698-3423<br>Fax: (214) 571-2961<br>Email: <a href="mailto:ncurtis@gibsondunn.com">ncurtis@gibsondunn.com</a> |

A Power of Attorney accompanies this Petition in accordance with 37 C.F.R. § 42.10(b).

**D. Service Information [37 C.F.R. § 42.8(b)(4)]**

Service via hand delivery or postal mail may be made at the address of the lead and back-up counsel above. Petitioner hereby consents to electronic service, and service by electronic mail may be made at the following email address:

- [GDC-DockerIPR@gibsondunn.com](mailto:GDC-DockerIPR@gibsondunn.com)

## **I. INTRODUCTION**

Pursuant to 35 U.S.C. §311, Petitioner petitions for *inter partes* review of U.S. Patent No. 8,332,844 (EX1001) and seeks cancellation of claims 1–27 as unpatentable under 35 U.S.C. §103.

The '844 patent generally relates to a block-level “branching store file system,” which maintains different versions of a file system for different users (compute nodes) by storing (1) “a read-only base image (or ‘root’ image),” and (2) one or more “leaf” images that record the changes to the “root” image unique to a compute node. EX1001, 2:13–18. When a user requests a copy of the file system, a “filter” between the compute nodes and the file systems “merges the changes recorded on the leaf images with the root image and delivers the result to the appropriate compute node.” *Id.*, 2:18–21. Thus, “[f]rom the point of view of the compute node, it is running its own unique and cohesive instance of the application environment.” *Id.*, 2:22–24.

The Challenged Claims are directed to this “root/leaf” system for storing blocks of data, in two categories: (1) the “caching” claims (1–13); and (2) the “indexing” claims (14–27). The “caching” claims relate to caching portions of the root image, whereas the “indexing” claims relate to indexing the root image by a compute node, and providing the resultant index to other compute nodes.

During prosecution, each of the independent claims was amended to recite that the “leaf images” comprise “only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of the root image.” EX1002, 166–71. The claims were allowed following the addition of this single allegedly novel feature. *Id.*, 279–83.

But, as explained in this Petition, this allegedly novel limitation—as well as the remaining claim limitations in the “caching” set of claims—would have been obvious over Menage as explained in Ground 1. Menage discloses the use of a set of shared storage units to store an initial file system (*i.e.*, a “root image” in the terminology of the ’844 patent) for a plurality of virtual private servers, and a set of private storage units that store changes to the initial file system made by each virtual process (*i.e.*, “leaf images”).

Similarly, the “caching” claims would have been obvious in light of the combination of Birse and Rothman, which is presented in Ground 3. Birse discloses shared system volumes that contain an operating system (*i.e.*, a “root image”) for a plurality of client computers and shadow volumes that store changes to the operating system (*i.e.*, “leaf images”), and Rothman teaches caching techniques for boot images used for network booting.

Moreover, Menage and Birse, each in combination with the Murphy reference (Grounds 2 and 4 herein), render obvious the “indexing” claims. Murphy describes

the indexing of an “image file” (*i.e.*, a “root image”) and the ability to provide the indexing results to another client computer by storing an “index file” on a shared server.

Petitioner respectfully requests the cancellation of the Challenged Claims.

## **II. REQUIREMENTS OF *INTER PARTES* REVIEW**

### **A. Standing**

Pursuant to 37 C.F.R. §42.104(a), Petitioner certifies that the ’844 patent is available for IPR and that Petitioner is not barred or estopped from requesting an IPR on the grounds identified herein.

### **B. Identification of Challenge and Relief Requested**

Pursuant to 37 C.F.R. §42.104(b), Petitioner requests that the Board institute IPR of claims 1–27 of the ’844 patent under pre-AIA 35 U.S.C. §103.

The ’844 patent issued from an application filed February 21, 2007, and is a continuation-in-part of an application filed on March 30, 2006, which is a continuation-in-part of an application filed on December 30, 2004. For purposes of this Petition only, Petitioner assumes that the Challenged Claims are entitled to an effective filing date of December 30, 2004.

The precise relief requested by Petitioner is that claims 1–27 be canceled based on the grounds below:

| <b>Ground</b> | <b>Claims</b> | <b>Basis for Rejection<sup>1</sup></b>    |
|---------------|---------------|-------------------------------------------|
| 1             | 1–13          | Obviousness over Menage                   |
| 2             | 14–27         | Obviousness over Menage in view of Murphy |
| 3             | 1–13          | Obviousness over Birse in view of Rothman |
| 4             | 14–27         | Obviousness over Birse in view of Murphy  |

The above challenges are made in view of the following prior art references:

**1. U.S. Patent No. 6,618,736 (“Menage”) (EX1004)**

U.S. Patent No. 6,618,736 (“Menage”), titled “Template-Based Creation and Archival of File Systems,” issued on September 9, 2003 from an application filed on March 9, 2001. Menage therefore qualifies as prior art under pre-AIA 35 U.S.C. §102(a) and (b). Menage was not considered during prosecution of the ’844 patent.

**2. U.S. Patent No. 7,395,324 (“Murphy”) (EX1005)**

U.S. Patent No. 7,395,324 (“Murphy”), titled “Method and Apparatus for Maintaining a Computer System,” issued on July 1, 2008 from an application filed on June 28, 2000. Murphy therefore qualifies as prior art under pre-AIA 35 U.S.C. §102(e). Murphy was not considered during prosecution of the ’844 patent.

**3. U.S. Patent No. 7,089,300 (“Birse”) (EX1006)**

U.S. Patent No. 7,089,300 (“Birse”), titled “Method and Apparatus for Administering the Operating System of a Net-Booted Environment,” issued on

---

<sup>1</sup> All obviousness grounds include the knowledge of POSITA.

August 8, 2006 from an application filed on October 18, 1999. Birse therefore qualifies as prior art under pre-AIA 35 U.S.C. §102(e). Birse was not considered during prosecution of the '844 patent.

**4. U.S. Patent No. 7,398,382 (“Rothman”) (EX1007)**

U.S. Patent Application Publication 7,398,382 (“Rothman”), titled “Method and Apparatus to Enhance Platform Boot Efficiency,” issued on July 8, 2008 from an application filed on December 29, 2004. Rothman therefore qualifies as prior art under pre-AIA 35 U.S.C. §102(e). Rothman was not considered during prosecution of the '844 patent.

**C. How the Challenged Claims Are To Be Construed Under 37 C.F.R. §42.104(b)(3)**

A claim subject to IPR shall be construed using the same standard as in a civil action. 37 C.F.R. §42.100(b). The Board need only construe terms “to the extent necessary to resolve the controversy.” *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017). Petitioner submits that no construction of any of any claim term is necessary for the Board to resolve. The Challenged Claims would have been obvious under any reasonable construction.

**D. How the Challenged Claims Are Unpatentable Under 37 C.F.R. §42.104(b)(4)**

An explanation of how the Challenged Claims are unpatentable under the statutory grounds identified above, including the identification of where each

element of the claim is found in the prior art patents or printed publications, is provided below.

**E. Supporting Evidence Under 37 C.F.R. §42.104(b)(5)**

The exhibit numbers of the supporting evidence relied upon and the relevance of the evidence to the challenged claims, including an identification of specific portions of the evidence that support the challenge, are provided below. A list of exhibits is also included in this petition pursuant to 37 C.F.R. §42.63(e). The technical information and grounds for rejection explained in the Petition are further supported by the Declaration of Dr. Erez Zadok (EX1003).

**F. Payment of Fees**

Pursuant to 37 C.F.R. §§42.103 and 42.15(a), the required fee is being submitted herewith. The Office is authorized to charge any fee deficiency, or credit overpayment, to Deposit Account 50-1408.

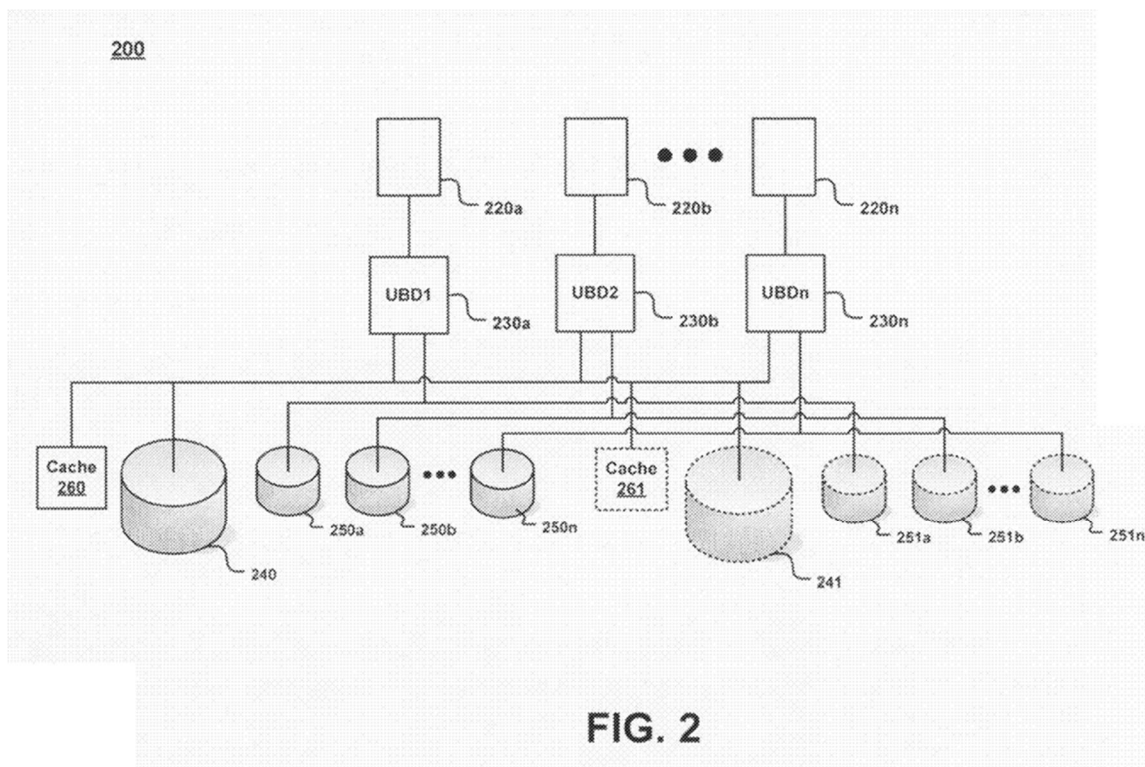
**III. TECHNICAL BACKGROUND**

**A. The '844 Patent and Summary of the Alleged Invention**

The '844 patent describes a system for “root image caching and indexing for block-level distributed application management.” EX1001, Abstract. However, the '844 patent is primarily directed to describing a block-level “branching store file system” for use in providing an “application environment” (*e.g.*, an operating system) to multiple “compute nodes” in a clustered computing environment.

In a “branching store file system,” a “read-only base image (or ‘root’ image) of the application environment is created,” and “is accessible by all compute nodes in the cluster.” *Id.*, 2:13–16. Further, “[c]hanges made by a compute node to the root image are stored in a ‘leaf’ image unique to that compute node.” *Id.*, 2:17–18. A filter “merges the changes recorded on the leaf images with the root image and delivers the results to the appropriate compute node.” *Id.*, 2:18–21. Thus, from the compute node’s point of view, “it is running its own unique and cohesive instance of the application environment.” *Id.*, 2:22–24.

Figure 2 shows an exemplary system “for root image caching and indexing in a block-level distributed application environment,” which depicts a “system 200” that “is implemented in a multi-computer system, such as an HPC cluster”:



*Id.*, 5:12–19, Fig. 2. “[C]ompute nodes 220*a-n*” are coupled to a “first storage unit 240” and a corresponding “second storage unit 250*a-n* through a corresponding union block device (UBD) 230*a-n*.” *Id.*, 5:19–22. To the compute nodes, “it appears that they have access to their own version of a distributed application environment,” but “a separate and complete boot image is not created and stored for each compute node.” *Id.*, 5:22–26.

The “first storage unit 240” stores “blocks of a root image of an application environment,” which “contains data initially common to the compute nodes” and “is not changed by [the] compute nodes,” *i.e.*, the compute nodes have read-only access to the “first storage unit.” *Id.*, 5:27–32. “[S]econd storage units 250*a-n*” each store a “leaf image,” which “may contain blocks of new data, blocks of changed data, or other blocks of data unique to the individual compute nodes,” and a “block modification log.” *Id.*, 5:33–45.

The Challenged Claims are directed to methods and systems for distributing an application environment to a compute node using a “root-leaf” system, where blocks of a root image are stored on a first storage unit, and “leaf images” of the compute nodes are stored on second storage units. *Id.*, 2:35–55.

## **B. Prosecution History**

The ’844 patent was filed with claims that broadly recited that the leaf images comprised “additional data blocks not previously contained in said root image and

changes made by respective compute nodes to the blocks of [said/the] root image,” or similar language. EX1002, 25–30. After a series of rejections, the claims were allowed based on the requirement that the leaf images comprise/include “only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of [said/the] root image.”

**C. Person of Ordinary Skill in the Art**

A POSITA in the field on or around December 30, 2004 would have had a Bachelor’s degree in Computer Science, Computer Engineering, or an equivalent degree, and at least two years’ worth of experience in the field of networked data storage, file systems, and operating systems. Additional education could substitute for experience, and vice versa. EX1003, ¶¶34–36.

**IV. GROUNDS OF UNPATENTABILITY**

**A. Ground 1: Obviousness over Menage (Claims 1–13)**

**1. Summary of Menage**

Menage relates to “file system creation and archival, particularly in the context of multiple virtual private servers running on a single physical host machine.” Menage explains that a “virtual private server is an example of a ‘virtual process,’ which is a set of processes isolated partially or totally from other processes of the system.” EX1004, 3:56–67.

Menage recognizes that one “difficulty in implementing multiple virtual private servers within a single physical host involves providing each server with a separate file system,” and explains that “providing a separate physical device for storing the file system of each virtual private server would be expensive and inefficient.” *Id.*, 7:30–44. Accordingly, “it would be desirable to store the file systems of multiple virtual private servers within the same physical device or comparatively small set of devices.” *Id.*, 7:44–47. Thus, Menage indicates that “what is needed is a technique for creating separate file systems for a plurality of virtual private servers that does not require extensive copying or wasted storage space.” *Id.*, 2:33–36.

Menage solves these problems “by providing a set of shared storage units 302 and a set of private storage units 304”:

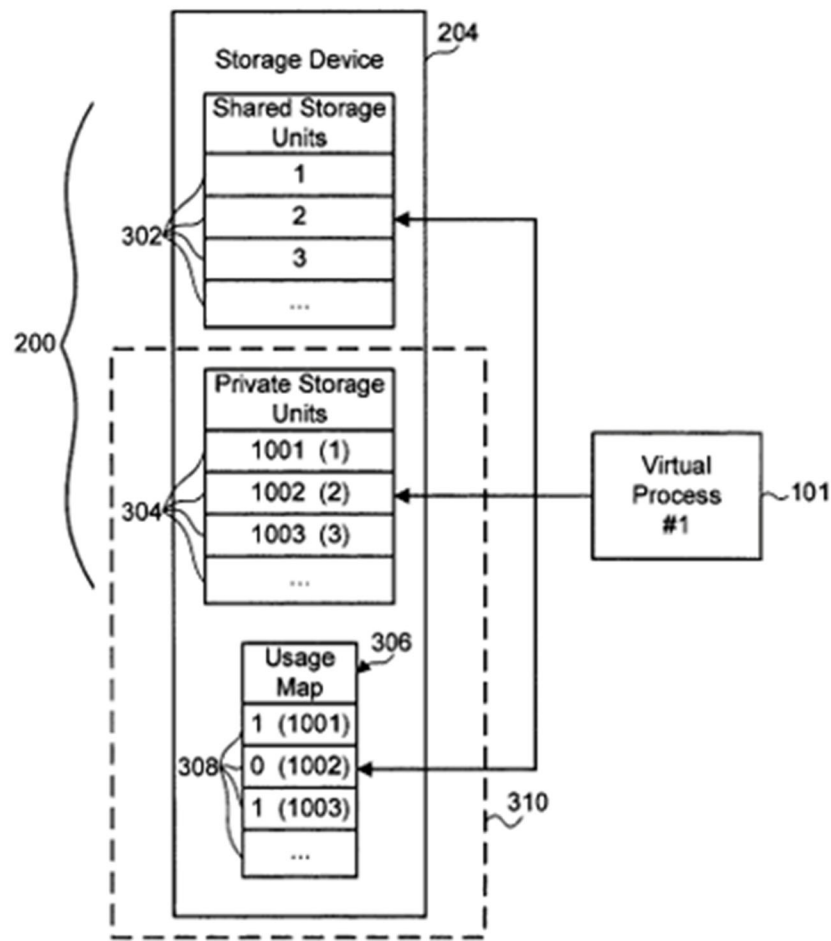


FIG. 3

*Id.*, 8:8–10, Fig. 3. The “set of shared storage units 302 is used to store data that is common to the files systems 200 of multiple virtual processes 101,” (*i.e.*, virtual private servers), “such as standard application programs, utilities, databases, etc.” *Id.*, 8:10–14. Further, “[t]he set of shared storage units 302 forms an initial file system 200 for each virtual process 101.” *Id.*, 8:14–15.

Conversely, “[t]he set of private storage units 304 ... are used to store changes to the initial file system 200 that occur subsequent to the creation of the file system

200.” *Id.*, 8:16–18. While the “private storage units 304” are depicted in Figure 3 “as being located within the same storage device 204 as the shared storage units 302,” they may also “be located within a separate storage device 204, either within the same physical server or within a remote server.” *Id.*, 8:22–27.

Menage also provides “a usage map 306” for each set of private storage units, which “includes a plurality of indicators 308 for indicating whether a corresponding private storage unit 304 ‘contains valid data,’” meaning “that the private storage unit 304 stores data representing a change to the initial file system 200. *Id.*, 8:28–34. Menage collectively refers to a set of private storage units 304 for a virtual process 101 and a corresponding usage map 306 as a “template,” and notes that “at least one template 310 is preferably provided for each virtual process 101.” *Id.*, 8:58–67.

Figures 4–6 illustrate Menage’s system for creating and managing the file systems 200, which uses a “system call wrapper” to intercept system calls to write data to or read data from the shared storage unit. Regarding write requests, the “system call wrapper” intercepts a system call for “writing a data item to a shared storage unit 202”:

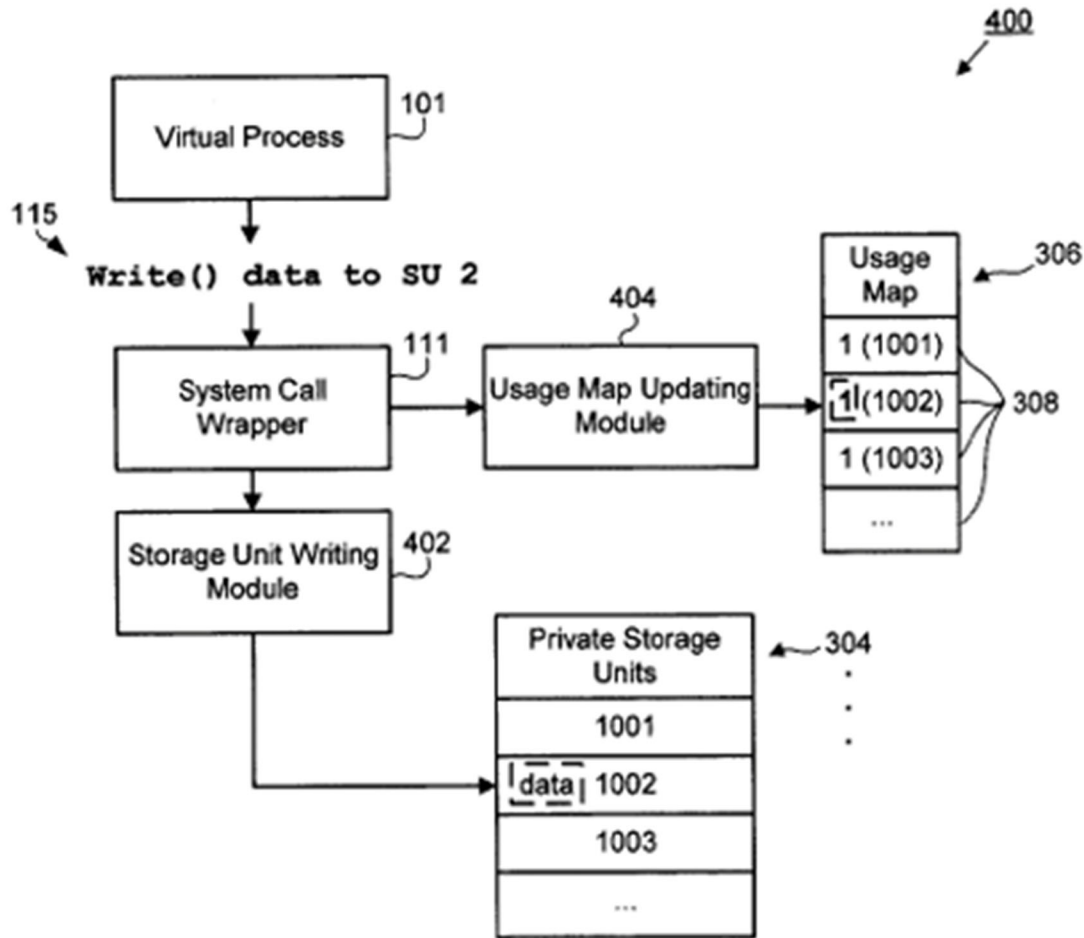
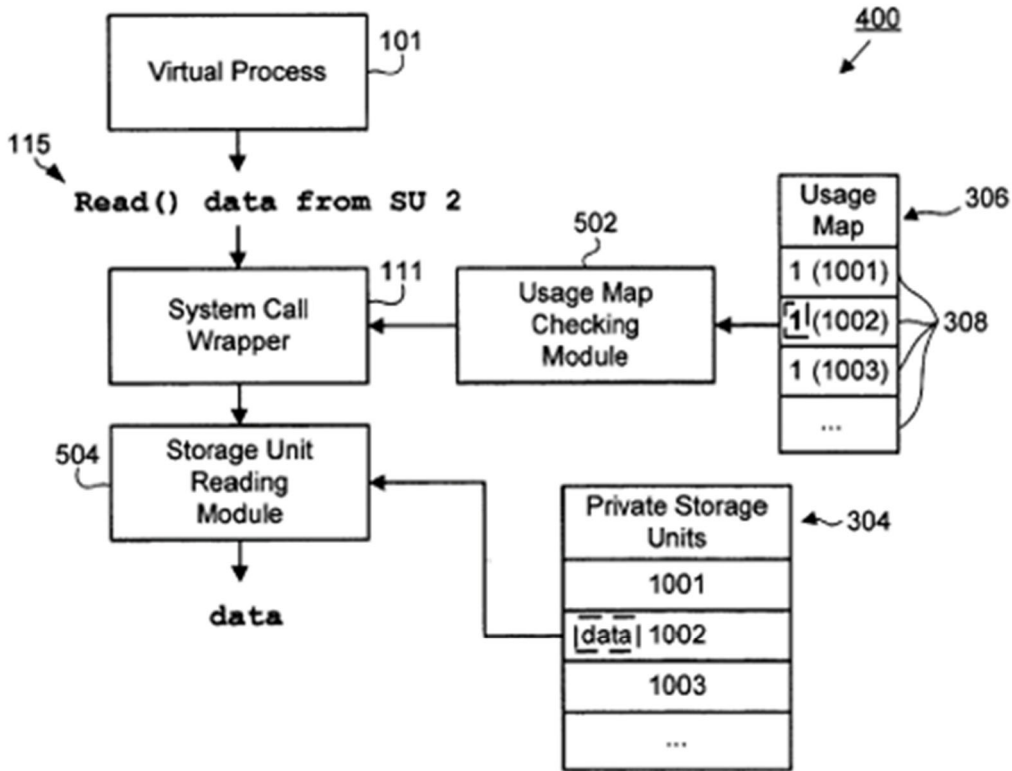


**FIG. 4**

*Id.*, 9:4–7, Fig. 4. Instead of writing the data item directly to the shared storage unit 302, “a storage unit writing module 402 writes the data item to the corresponding private storage unit 304,” and “a usage map updating module 404 stores an indication 308 in the usage map 306 that the private storage unit 304 (to which the data item is written) contains valid data.” *Id.*, 9:10–21.

Regarding read requests, the “system call wrapper” intercepts a system call for “reading a data item from a shared storage unit 302,” as shown in Figure 5:



*Id.*, 9:29–32, Fig. 5. Instead of reading the data item directly from the shared storage unit 302, “a usage map checking module 502 checks for an indication 308 in the usage map 306 that the corresponding private storage unit 304 contains valid data,” and if it does, “a storage unit reading module 504 reads the data item from the private storage unit 304 rather than the shared storage unit 302.” *Id.*, 9:35–41. Conversely, if “the usage map 306 does not include an indication 308 of valid data for the

corresponding private storage unit 304,” than the data item is read from the shared storage unit.

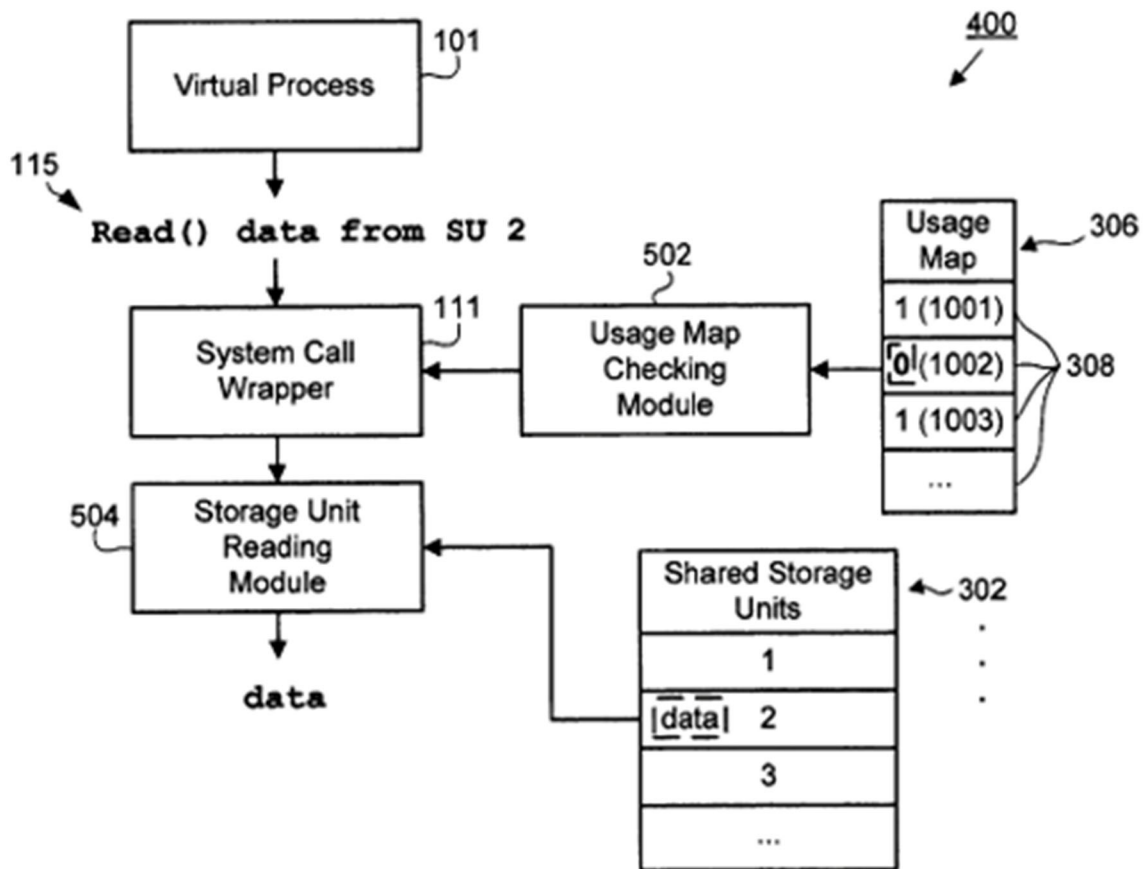


FIG. 6

*Id.*, 9:48–53; Fig. 6.

“[T]he file system 200 of each virtual process 101 may be conceptualized as a combination of the set of shared storage units 302 and the set of private storage units 304 from the corresponding template 310”:

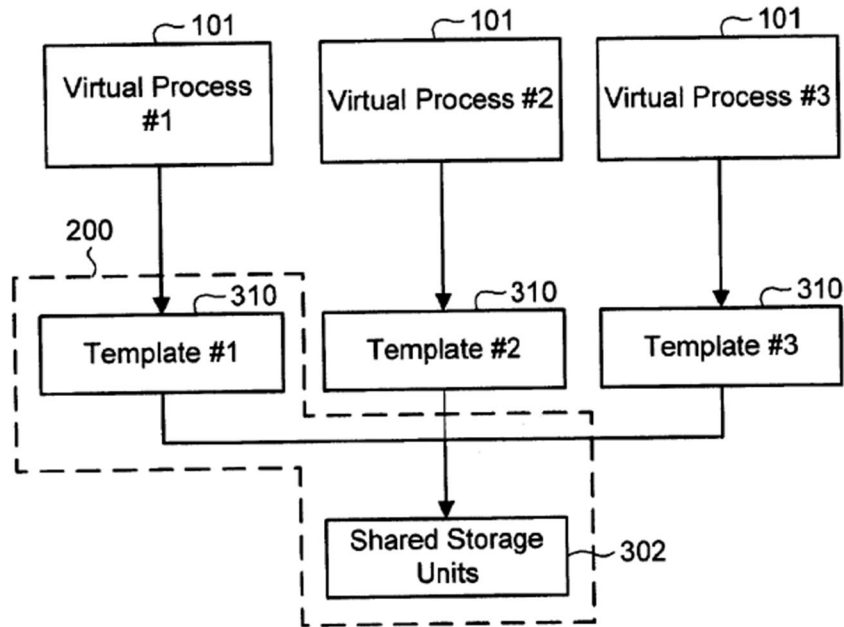


**FIG. 7**

*Id.*, 9:54–57, Fig. 7.

In addition to the advantage of not needing to duplicate an initial file system for each virtual process, in Menage, “since all the virtual processes 101 use the same set of shared storage units 302, the shared storage units may be easily cached for high speed access in the computer memory 102 or in a memory integrated with the storage device 204.” *Id.*, 10:2–6.

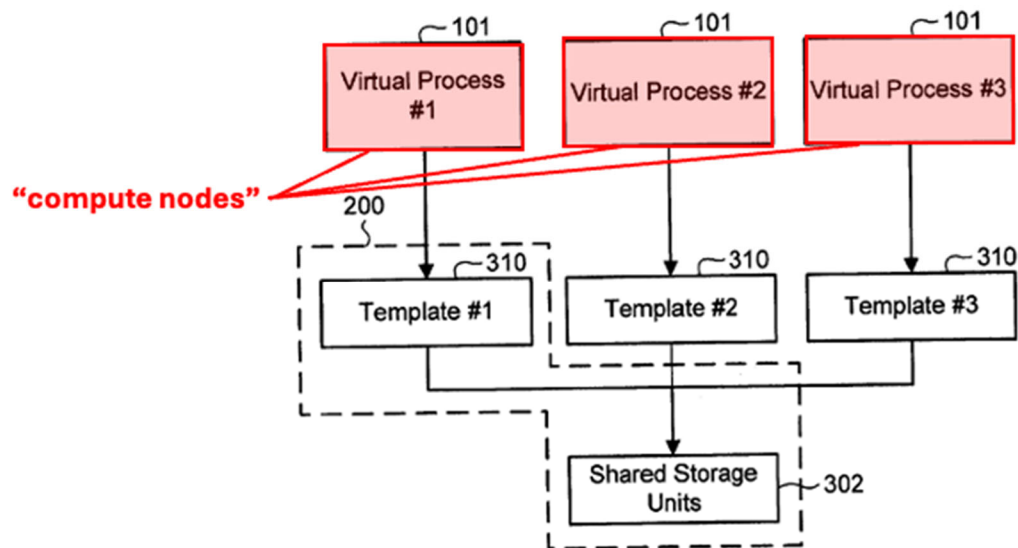
Menage discloses and/or renders obvious claims 1–13 of the ’844 patent.

## 2. Detailed Analysis of Ground 1

### a. Independent Claim 1

- 1) [1.pre] “A system for providing data to a plurality of compute nodes, the system comprising:”

If claim 1’s preamble is limiting, Menage discloses and/or renders it obvious. EX1003, ¶¶296–301. Menage “relates to file system creation ... particularly in the context of multiple virtual private servers running on a single physical host machine.” EX1004, 3:56–58. “[T]he file system 200 of each virtual process 101” (*i.e.*, “**a plurality of compute nodes**”) is “a combination of the set of shared storage units 302 and the set of private storage units 304 from the corresponding template 310”:



*Id.*, 10:54–57, Fig. 7.<sup>2</sup>

<sup>2</sup> All coloring and annotations are added unless otherwise noted.

In the '844 patent, the embodiments describing “compute nodes” are exemplary and there is nothing excluding a virtual process on a “virtual private server” from being a “compute node” within the meaning of the '844 patent. *E.g.*, EX1001, 1:35–37, 3:25–27, 4:26 (“Example Compute Node Operating Environment”). A POSITA would therefore have understood “compute node” broadly to include both physical and virtual nodes. EX1003, ¶297. Thus, Menage teaches “a plurality of compute nodes” through its virtual processes.

To the extent Patent Owner argues that a “compute node” in the claims of the '844 patent must be a separate, physical node, Menage discloses and/or renders obvious [1.pre] under that narrow interpretation. Menage explains, for example, that it may be desirable to spread “virtual private servers” across a “comparatively small set of devices.” EX1004, 2:3–6, 7:42–47. In that embodiment, each “device” running a virtual process would be a “compute node.”

It further would have been obvious to a POSITA that the “virtual processes” of Menage could have been distributed across multiple, physical nodes. EX1003, ¶300. Doing so would achieve both scalability and fault tolerance. Distributing virtual servers enables horizontal scaling, allowing the overall system to handle increased workloads by leveraging the combined resources of multiple physical machines. This approach not only improves performance but also enhances system resilience; if one physical node fails, virtual servers can be migrated or restarted on

other available nodes, minimizing downtime and ensuring continued availability of services. *Id.*

Accordingly, Menage teaches and/or renders obvious [1.pre]. EX1003, ¶301.

**2) [1.a] “a first storage unit configured to store blocks of a root image of said compute nodes;”**

Menage discloses and/or renders this limitation obvious. EX1003, ¶¶302–06. Menage’s “shared storage units 302” are “a first storage unit configured to store blocks of a root image of said compute nodes.” *Id.*, ¶302. The “set of shared storage units 302” (*i.e.*, “**a first storage unit**”) “is used to store data that is common to the files systems 200 of multiple virtual processes 101, such as standard application programs, utilities, databases, etc.,” and “forms an initial file system 200” (*i.e.*, “a root image of said compute nodes”) “for each virtual process 101.” The set of data common to multiple processes or nodes is a “root image” as that term is used in the ’844 patent. *Id.*

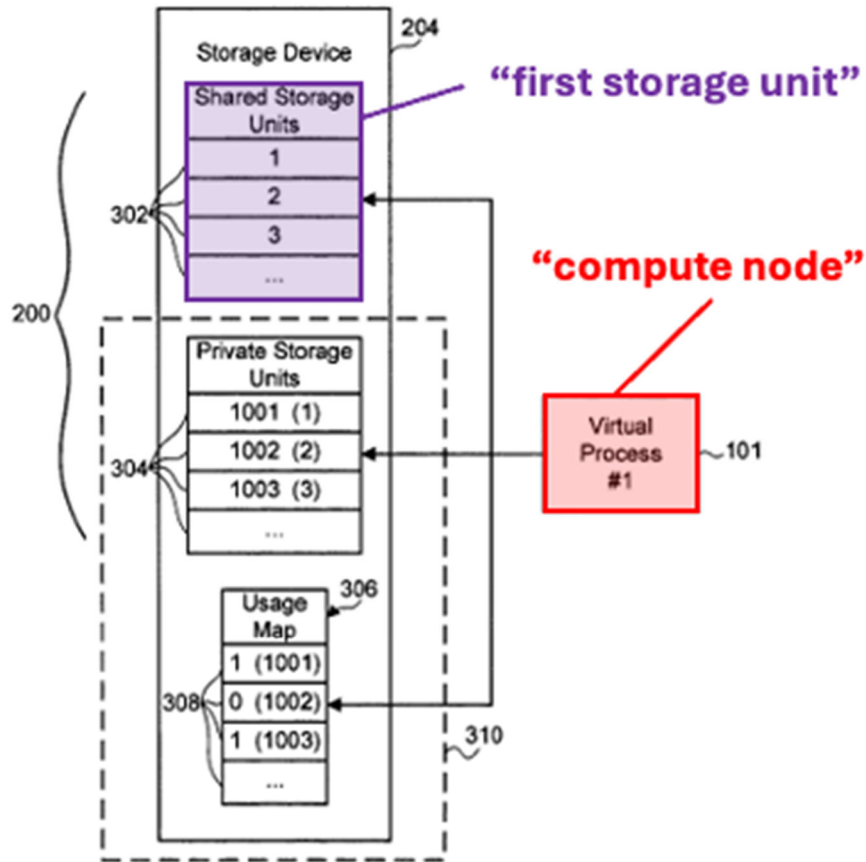
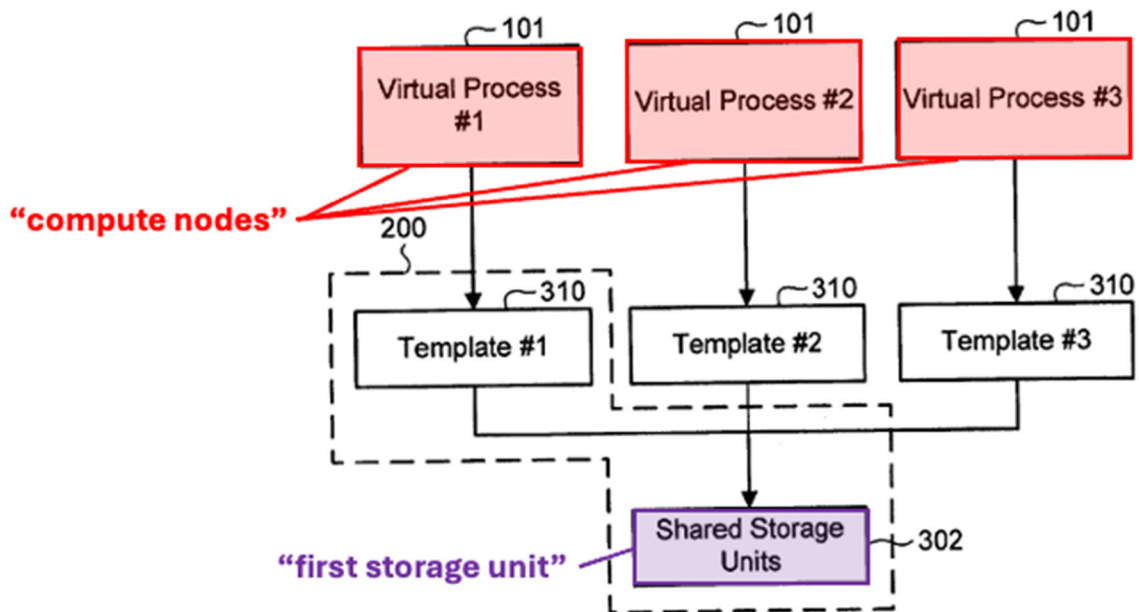


FIG. 3



*Id.*, 8:10–15, Figs. 3, 7.

Menage explains that data stored in a file system “is typically organized into ‘files,’” that “are stored within a plurality of ‘storage units’ of the storage device, sometimes referred to as ‘disk blocks’ or ‘allocation units.’” *Id.*, 7:33–40. Thus, the first storage unit (*i.e.*, storage units 302) store *blocks* of a “root image” of Menage (*i.e.*, the data that is common to the files systems 200 of multiple virtual processes 101). EX1003, ¶¶304–05.

Accordingly, Menage teaches [1.a]. EX1003, ¶306.

**3) [1.b.i] “a plurality of second storage units configured to store leaf images of respective compute nodes,”**

Menage discloses and/or renders this limitation obvious. EX1003, ¶¶307–09. Menage’s “templates,” which include “a set of private storage units 304 for a virtual process 101 and a corresponding usage map 306” are “a plurality of second storage units configured to store leaf images of respective compute nodes.” The “set of private storage units 304” in each template (*i.e.*, “**a plurality of second storage units**”) “are used to store changes to the initial file system 200 that occur subsequent to the creation of the file system 200” (*i.e.*, “leaf images of respective compute nodes”):

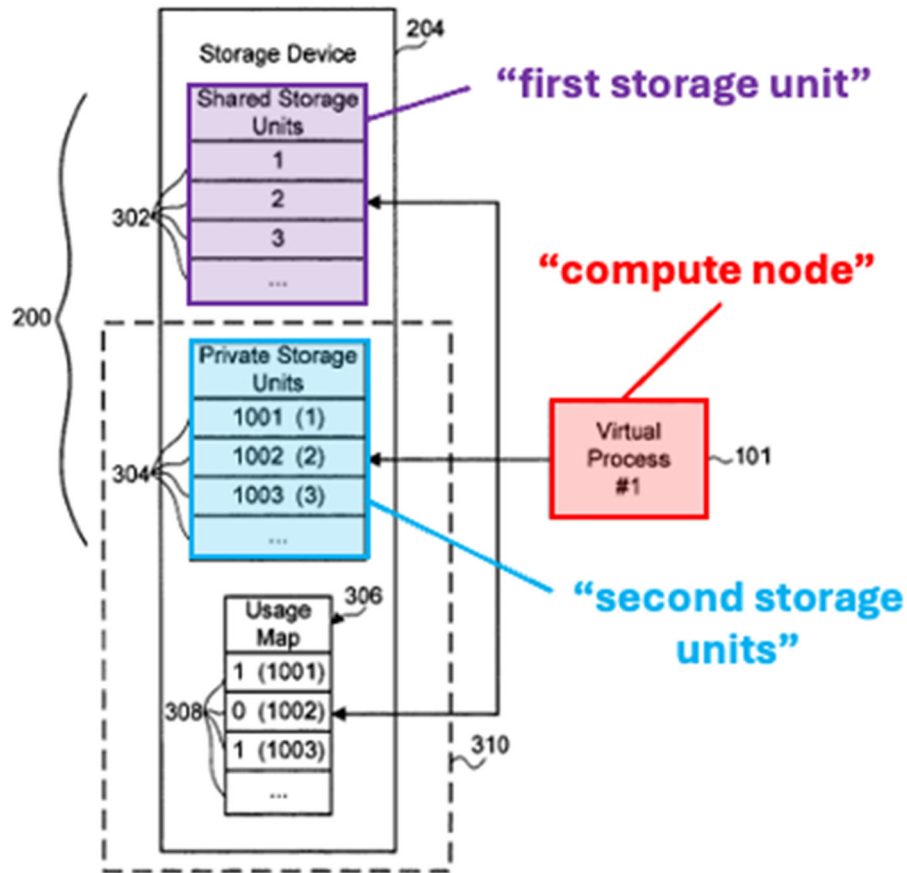
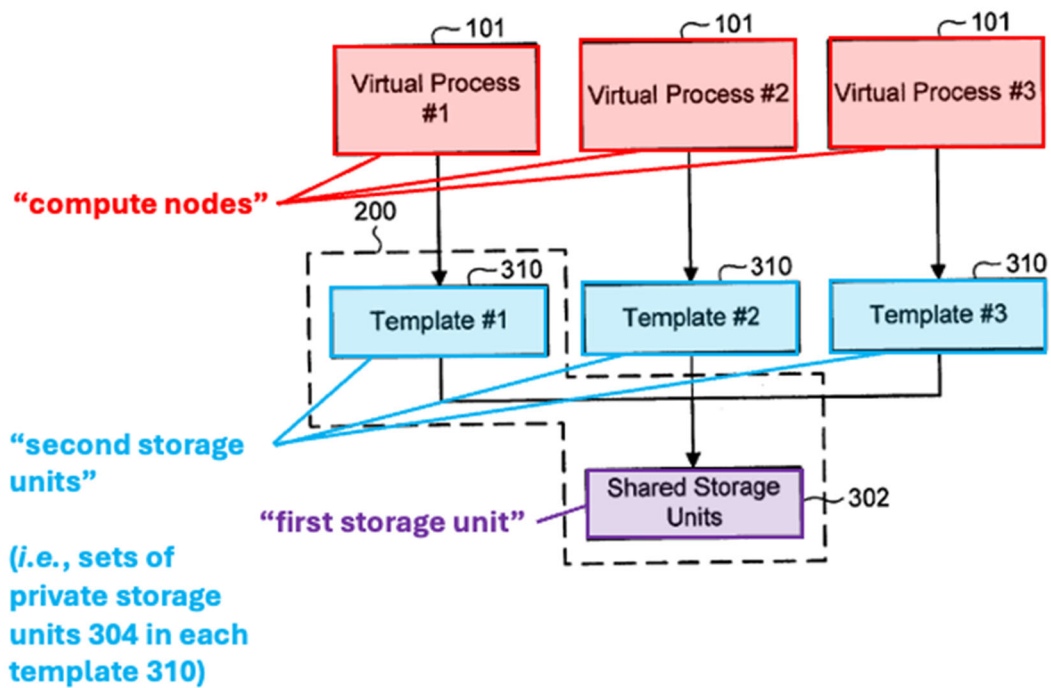


FIG. 3



*Id.*, 8:15–17, Figs. 3, 7. Moreover, “at least one template 310 is preferably provided for each virtual process 101.” *Id.*, 8:61–67. Thus, the “set of private storage units 304” in each template in Menage (the claimed “leaf images”) are maintained for “respective compute nodes.” EX1003, ¶308.

Accordingly, Menage teaches [1.b.1]. *Id.*, ¶309.

- 4) **[1.b.ii] “said leaf images including only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of said root image, wherein said leaf images of respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes; and”**

Menage discloses and/or renders this limitation obvious. EX1003, ¶¶310–19. Menage explains that “[t]he set of private storage units 304” (*i.e.*, the “leaf images”) “are *used to store changes to the initial file system 200*” (*i.e.*, the “root image”) “that occur subsequent to the creation of the file system 200.” EX1004, 8:16–18.

According to Menage, “[a]t least one template 310,” each of which includes “a set of private storage units 304 for a virtual process 101 and a corresponding usage map 306,” is provided for each virtual process. *Id.*, 8:58–61. Menage also explains that “the usage map 306 is initially reset or ‘initialized’ to indicate that none of the private storage units 304 contain valid data,” with the phrase “contains valid data” meaning “that the private storage unit 304 stores data representing a change to the

initial file system 200.” *Id.*, 8:28–34, 8:46–49. There is no suggestion in Menage that a “template” would store a duplicate copy of data blocks also in the “root image” in Menage. EX1003, ¶318.

When a virtual process (*i.e.*, a “compute node”) makes a system call to write data to the shared storage unit 302, the system call is intercepted by a “system call wrapper 111”:

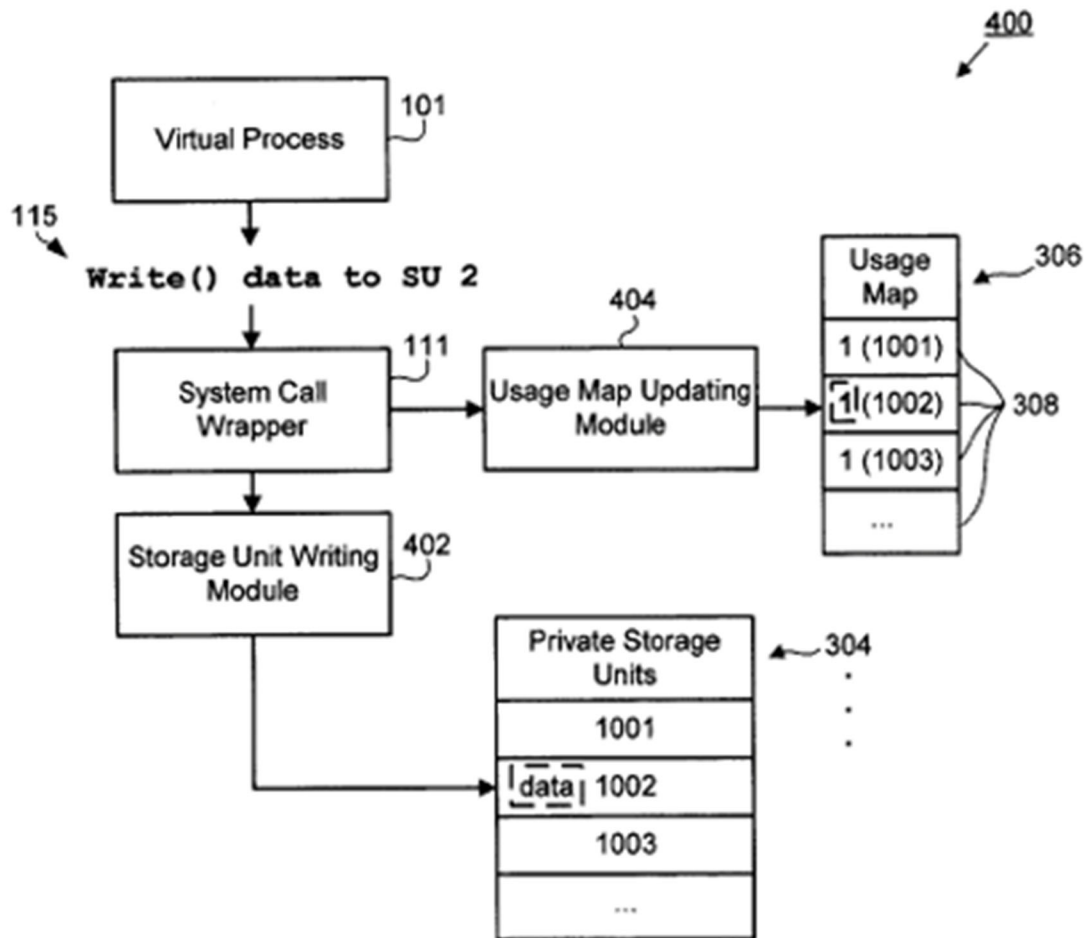


FIG. 4

*Id.*, 9:4–7, Fig. 4. “Rather than writing the data item to the shared storage unit 302, however, a storage unit writing module 402 *writes the data item to the corresponding private storage unit 304,*” and “a usage map updating module 404 stores an indication 308 in the usage map 306 that the private storage unit 304 (to which the data item is written) contains valid data.” *Id.*, 9:10–12, 18–23.

When a virtual process makes a system call to read data from the shared storage units 302, the system call is intercepted by the “system call wrapper 111,”:

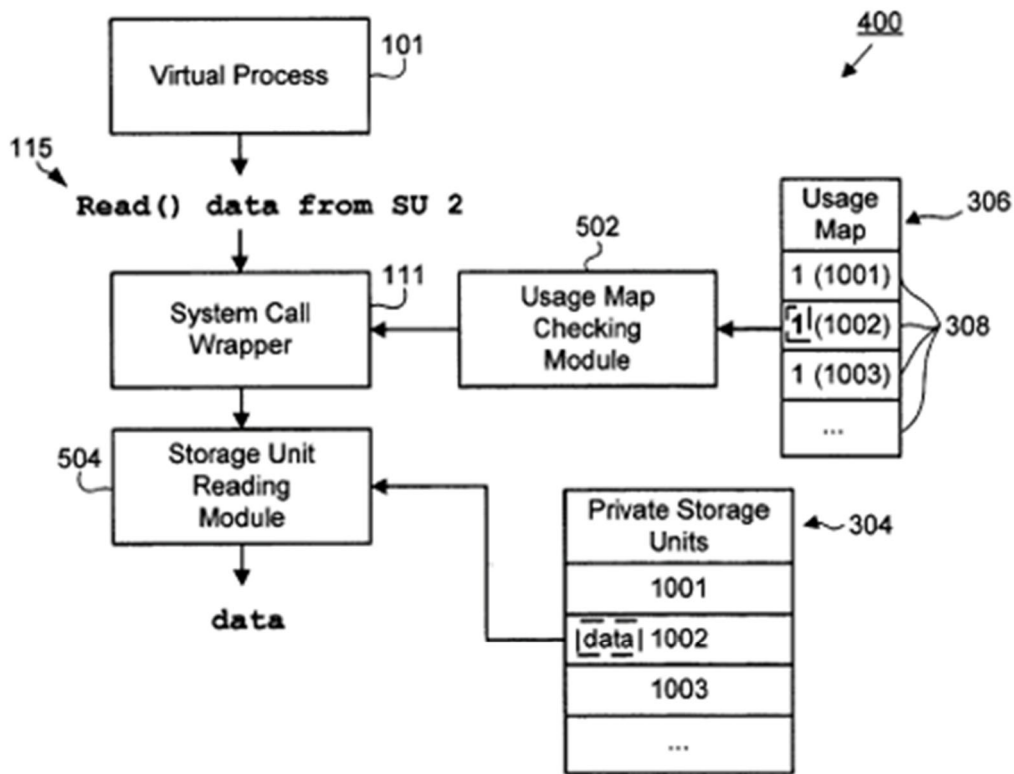


FIG. 5

*Id.*, 9:30–33, Fig. 5. Rather than reading the data item directly from the storage unit 302, “a usage map checking module 502 checks for an indication 308 in the usage map 306 that the corresponding private storage unit 304 contains valid data,” and if it does, “a storage unit reading module 504 reads the data item from the private storage unit 304 rather than the shared storage unit 302.” *Id.*, 9:35–41. Alternatively, “if the usage map checking module 502 determines that the usage map 306 does not include an indication 308 of valid data for the corresponding private storage unit 304, then the storage unit reading module 504 reads the data item from the shared storage unit 302 specified in the intercepted system call 115” (below):

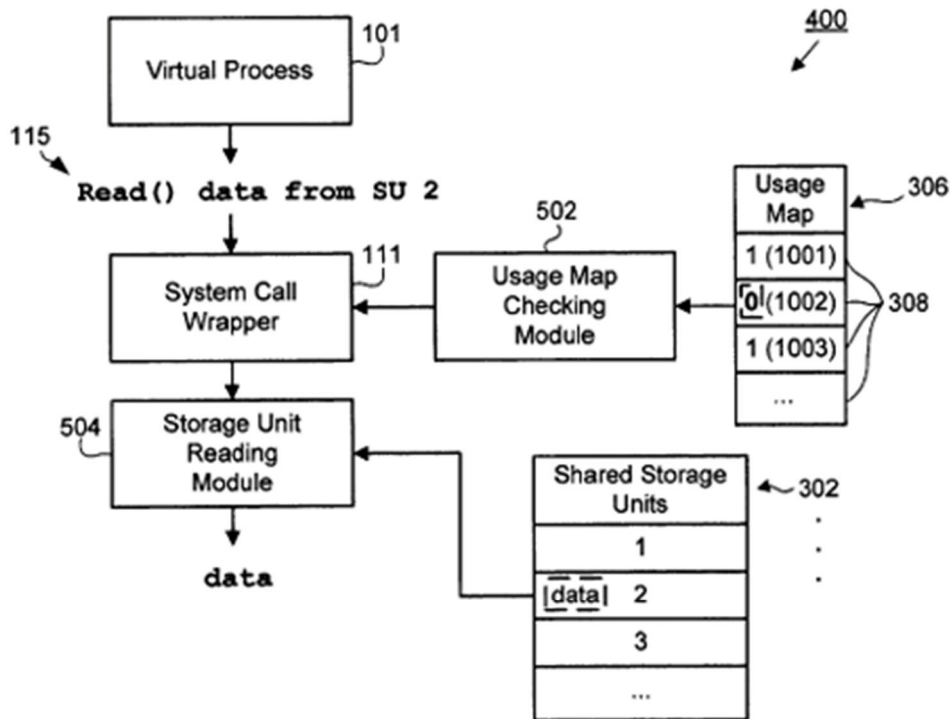


FIG. 6

*Id.*, 9:48–53, Fig. 6. Thus, data items that are unchanged by a virtual process (*i.e.*, the usage map does not indicate valid data) remain in the shared storage unit, and are not included in the private storage unit. EX1003, 318.

Accordingly, Menage teaches [1.b.ii]. *Id.*, ¶319.

**5) [1.c] “a cache configured to cache blocks of said root image previously accessed by at least one of said compute nodes.”**

Menage discloses and/or renders this limitation obvious. EX1003, ¶¶320–22. Menage explains that because “all the virtual processes 101 use the same set of shared storage units 302”—which, as discussed with respect to [1.a], store blocks of a root image of said compute nodes—“the shared storage units 302 *may be easily cached* for high speed access in the computer memory 102 or in a memory integrated with the storage device 204.” EX1004, 10:2–6. Moreover, it would have been obvious to a POSITA that blocks of the root image stored in the shared storage units 302 would have been cached, as described by Menage, upon read access by one of the virtual processes (*i.e.*, “one of said compute nodes”), in accordance with well-known caching techniques. EX1003, ¶321; EX1009, 42–43, 319–25.

Accordingly, Menage teaches [1.c]. EX1003, ¶322.

**b. Claim 2**

**“The system as recited in claim 1 wherein said cache is configured to store X most recently accessed blocks of said root image, and wherein X represents a cache threshold value.”**

Menage, in light of the knowledge of a POSITA, renders this element obvious. EX1003, ¶¶323–33. It would have been obvious a POSITA that this limitation merely describes a standard, well-known cache replacement policy known as “least recently used,” or “LRU.” In LRU, the least recently used data item is discarded from the cache first to make room for new data when the cache is full. By discarding the least recently used item from the cache, the most recently accessed items are stored. Moreover, the “cache threshold value” of “X” required by this element is nothing more than the size of the cache, either by hardware (*i.e.*, the physical capacity of the cache memory) or software (*i.e.*, the size of a portion of memory allocated to be used as cache). *Id.*, ¶¶325–32; EX1009, 20–23 (describing the LRU algorithm in an undergraduate operating systems textbook published in 1994).

Moreover, the ’844 patent recognizes that LRU is one “cache arbitration algorithm” that was “known in the art.” EX1001, 7:8–16.

Thus, Menage, in light of the knowledge of a POSITA, teaches claim 2. EX1003, ¶333.

c. Claim 3

**“The system as recited in claim 1 wherein said first storage unit, said second storage units, and said cache are contained within a single storage appliance.”**

Menage discloses and/or renders this limitation obvious. EX1003, ¶¶334–36.

Menage explains that “[t]he sets of storage units,” *i.e.*, the shared storage units 302 (*i.e.*, “first storage unit”) and private storage units 304 (*i.e.*, “second storage units”) may be “located *within the same storage device*”:

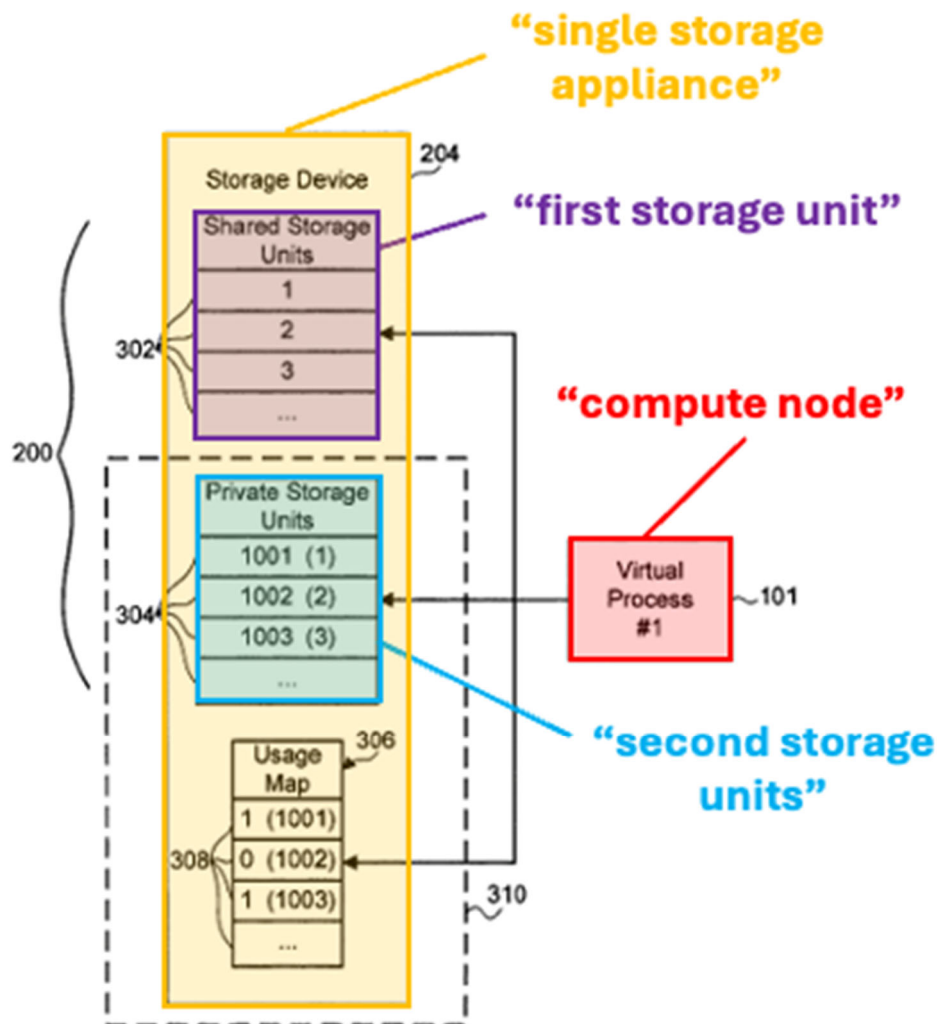


FIG. 3

EX1004, 8:22–24; Fig. 3; *see id.*, 7:50–53 (“[E]ach file system 200 is stored in a linear set of equal-sized storage units 202 within the same physical storage device.”). Likewise, Menage states that “the shared storage units 302 may be easily cached ... in a memory *integrated with the storage device 204.*” *Id.*, 10:2–6.

Thus, Menage teaches claim 3. EX1003, ¶336.

**d. Claim 4**

**“The system as recited in claim 1 further comprising:  
a plurality of union block devices configured to  
interface between respective compute nodes and said  
first storage unit, respective second storage units, and  
said cache to distribute application environments to  
the compute nodes, wherein said union block devices  
are configured to create said application environments  
by merging the blocks of said root image with the  
blocks of respective leaf images.”**

Menage discloses and/or renders these limitations obvious. EX1003, ¶¶337–41. Menage teaches the use of a system call wrapper, a storage unit writing module, a storage unit reading module, and a usage map checking module to intercept read and write requests to the shared storage units and redirect the operations to/from the private storage units as necessary. *See* [1.b.ii]. Although the ’844 patent notes that in one embodiment, “union block devices” or “UBDs” may be “low-level drivers,” they are not so limited in claim 4. *See* EX1001, cl. 5 (“The system as recited in claim 4 wherein said union block devices comprise low-level drivers.”). Thus, it would have been obvious to a POSITA that the system call wrapper and associated

modules are “a plurality of union block devices configured to interface between respective compute nodes and said first storage unit, respective second storage units, and said cache.”

Furthermore, the system call wrapper and associated modules “distribute application environments to the compute nodes” as required by this claim. EX1003, ¶340. The ’844 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.” EX1001, 2:14–18. Additionally, “[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.” *Id.*, 2:18–24.

The Menage system operates exactly the same way: “the set of shared storage units 302 is used to store data that is common to the files systems 200 of multiple virtual processes 101, such as standard application programs, utilities, databases, etc.,” and “forms an initial file system 200 for each virtual process 101.” EX1004, 8:10-15. On the other hand, the “set of private storage units 304 ... are used to store changes to the initial file system 200 that occur subsequent to the creation of the file

system 200,” and the system call wrapper and associated modules operate, using the usage map 306 for each set of private storage units, to “merge[] the changes” recorded on the private storage units (*i.e.*, “the blocks of respective leaf images”) with the “initial file system” stored in the set of shared storage units (*i.e.*, “the blocks of said root image”). *Id.*, 8:16–18; EX1003, ¶339.

Thus, Menage teaches claim 4. EX1003, ¶341.

**e. Claim 5**

**“The system as recited in claim 4 wherein said union block devices comprise low-level drivers for interfacing between the file systems of respective compute nodes and said first storage unit, respective second storage units, and said cache.”**

Menage discloses and/or renders these limitations obvious. EX1003, ¶¶342–46. Menage teaches the use of a system call wrapper, a storage unit writing module, a storage unit reading module, and usage map updating/checking modules to intercept read and write requests to the shared storage units and redirect the operations to/from the private storage units as necessary. *See* [1.b.ii] and claim 4.

Menage explains that “interception of system calls is illustrated as a primary vehicle for file system virtualization,” but that “virtualization *at other levels* is also contemplated.” For example, “virtualization may take place *at a lower level*, such as within a *virtual disk service*.” And, “[i]n such an embodiment, the virtualization

may occur once the file system code has turned a system call request into requests for particular blocks on the *virtual block device*.” EX1004, 12:15–22.

Accordingly, Menage teaches claim 5. EX1003, ¶346.

**f. Claim 6**

**“The system as recited in claim 1 wherein said first storage unit is read-only.”**

Menage discloses and/or renders this limitation obvious. EX1003, ¶¶347–50. System calls to write data to the shared storage unit 302 (*i.e.*, “said first storage unit”) are intercepted by a system call wrapper, and “[r]ather than writing the data item to the shared storage unit 302, however, a storage unit writing module 402 *writes the data item to the corresponding private storage unit 304*,” and “a usage map updating module 404 stores an indication 308 in the usage map 306 that the private storage unit 304 (to which the data item is written) contains valid data.” *See* [1.b.ii]; EX1004, 9:4–12. Thus, because all write requests directed to the shared storage unit 302 are redirected to the private storage unit 304, the shared storage unit 302 is “read-only” as required by claim 6. EX1003, ¶349.

Accordingly, Menage teaches claim 6. *Id.*, ¶350.

**g. Independent Claim 7**

**1) [7.pre] “A method for providing data to a plurality of compute nodes, comprising:”**

If claim 7’s preamble is limiting, Menage discloses and/or renders it obvious as discussed for as discussed for [1.pre] . EX1003, ¶351.

- 2) [7.a] “storing blocks of a root image of said compute nodes on a first storage unit;”**

Menage discloses and/or renders this limitation obvious as discussed for [1.a].

EX1003, ¶352.

- 3) [7.b.i] “storing leaf images for respective compute nodes on respective second storage units,”**

Menage discloses and/or renders this limitation obvious as discussed for [1.b.i]. EX1003, ¶353.

- 4) [7.b.ii] “said leaf images including only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of the root image, wherein said leaf images of respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes; and”**

Menage discloses and/or renders this limitation obvious as discussed for [1.b.ii]. EX1003, ¶354.

- 5) [7.c] “caching blocks of said root image that have been accessed by at least one of said compute nodes in a cache memory.”**

Menage discloses and/or renders this limitation obvious as discussed for [1.c].  
EX1003, ¶355.

**h. Claim 8**

**“The method as recited in claim 7 further comprising: receiving a read request from at least one of said compute nodes, wherein a first portion of the data requested is currently stored in said cache memory; and providing said first portion of said data to said at least one of said compute nodes from said cache memory.”**

Menage discloses and/or renders these limitations obvious. EX1003, ¶¶356–59. It would have been obvious to a POSITA that these limitations describe nothing more than the standard operation of a cache memory. Menage explains that “the shared storage units 302 may be easily cached *for high speed access* in the computer memory 102 or in a memory integrated with the storage device 204.” EX1004, 10:2–6. This “high speed access” is achieved because read requests for data items that are stored in the cache memory (*i.e.*, cache hits) are served from the cache, which is faster than the underlying data store. EX1003, ¶¶357–58.

Accordingly, Menage teaches claim 8. *Id.*, ¶359.

**i. Claim 9**

**“The method as recited in claim 8 further comprising: updating said cache memory based on said read request.”**

Menage discloses and/or renders this limitation obvious. EX1003, ¶¶360–63. Like claim 8, it would have been obvious to a POSITA that this limitation describes nothing more than the standard operation of a cache. *Id.*, ¶362. In the event of a cache miss (*i.e.*, when the cache is checked during a read request and the requested

data item is not stored in the cache), the data item is retrieved from the slower, underlying storage medium, and added to the cache memory for faster retrieval in response to a subsequent read request. Thus, it would have been obvious to a POSITA that any standard cache would have operated by “updating said cache memory based on said read request” when a data item is not stored in the cache memory. *Id.*

Accordingly, Menage teaches claim 9. *Id.*, ¶363.

**j. Claim 10**

**“The method as recited in claim 9 wherein a second portion of the data requested is not currently stored in said cache memory and said updating comprises: caching said second portion in said cache memory; and removing the least recently accessed data from said cache memory if the amount of data in said cache memory is above a threshold value”**

Menage discloses and/or renders these elements obvious as discussed for claims 2 and 9. EX1003, ¶364.

**k. Claim 11**

**“The method as recited in claim 7 further comprising: merging the blocks of said root image with the blocks of respective leaf images to create cohesive respective application environments.”**

Menage discloses and/or renders this element obvious as discussed for claim 4. EX1003, ¶365.

**l. Claim 12**

**“The method as recited in claim 11 wherein said merging occurs at an operational level between file systems of the respective compute nodes and said first storage unit, respective second storage units, and said cache memory.”**

Menage discloses and/or renders this element obvious as discussed for claim

5. EX1003, ¶366.

**m. Claim 13**

**“The method as recited in claim 7 wherein said first storage unit is read-only.”**

Menage discloses and/or renders this element obvious as discussed for claim

6. EX1003, ¶367.

**B. Ground 2: Obviousness over Menage in view of Murphy  
(Claims 14–27)**

**1. Summary of Murphy**

Murphy relates to “systems for installing and maintaining files, file systems, BIOS data, operating system data and other data on computers in a networked environment.” EX1005, 1:10–14. Specifically, Murphy discloses an administration system to “administer, maintain and upgrade computers from a central location,” which obviates the need for “support personnel to physically go to [an individual] computer” to perform administrative tasks, and for an administrator “to physically make changes or upgrades on each individual PC” when an upgrade or configuration change for a multiple PCs is necessary. *Id.*, 1:63–2:14.

In Murphy's remote management and configuration system, "one or more client computers 202" communicate with a "server 206 to obtain configuration information," which may include "administrative tasks that may be carried out by scripts provided by server 206 to client 202," such as "booting the operating system directly from server 206" or from "another server or a local drive":

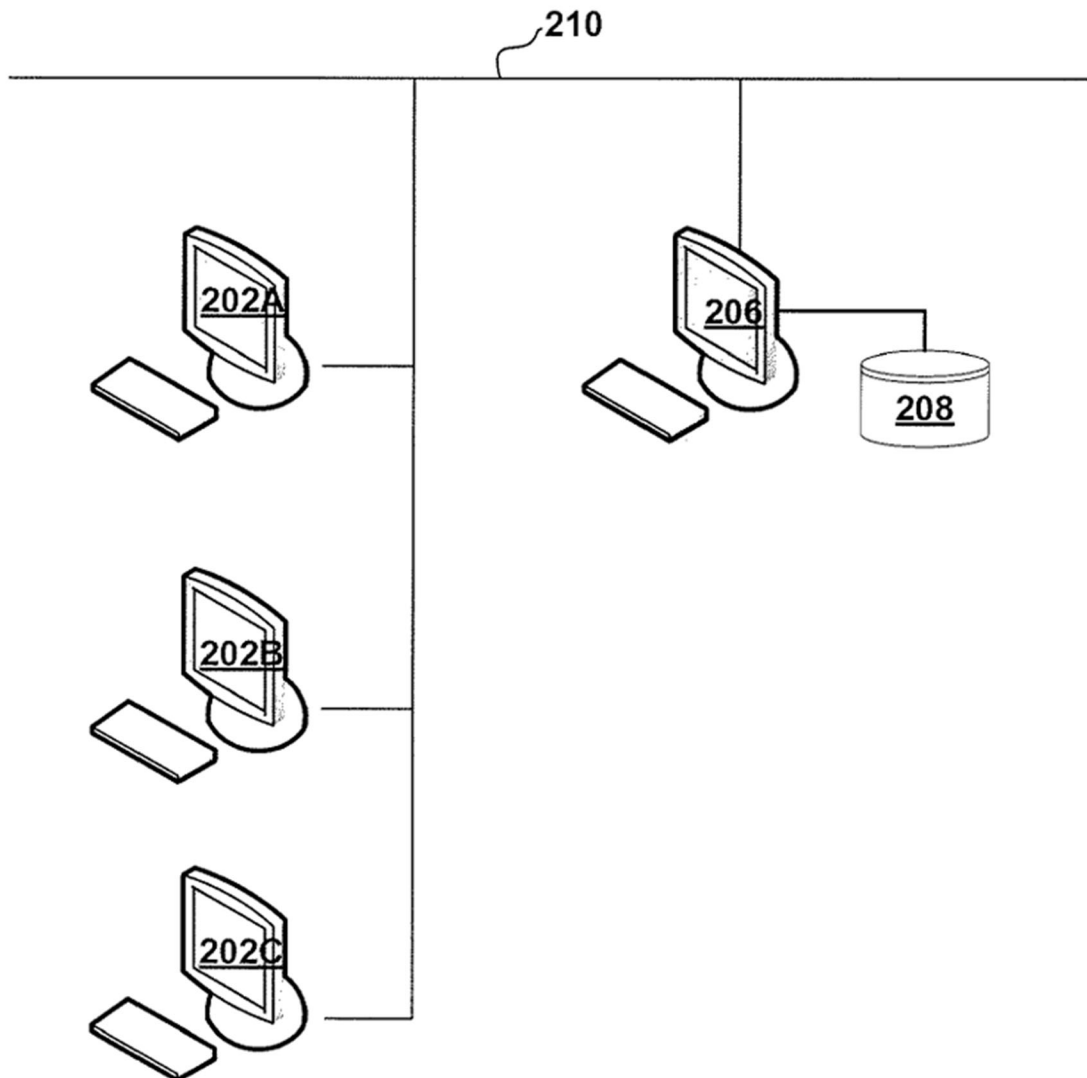


Figure 2

*Id.*, 3:41–4:12, Fig. 2.

Murphy describes a “boot image” that “may be retrieved from server 206” by a client computer 202. *Id.*, 11:37–39. The “boot image suitably includes an operating system kern[e]l, a command interpreter and one or more configuration scripts,” as well as “[v]arious other programs that may be included within the boot image,” such as “a file management program 412.” *Id.*, 11:55–59, 14:16–19. An exemplary boot image is depicted in Fig. 5:

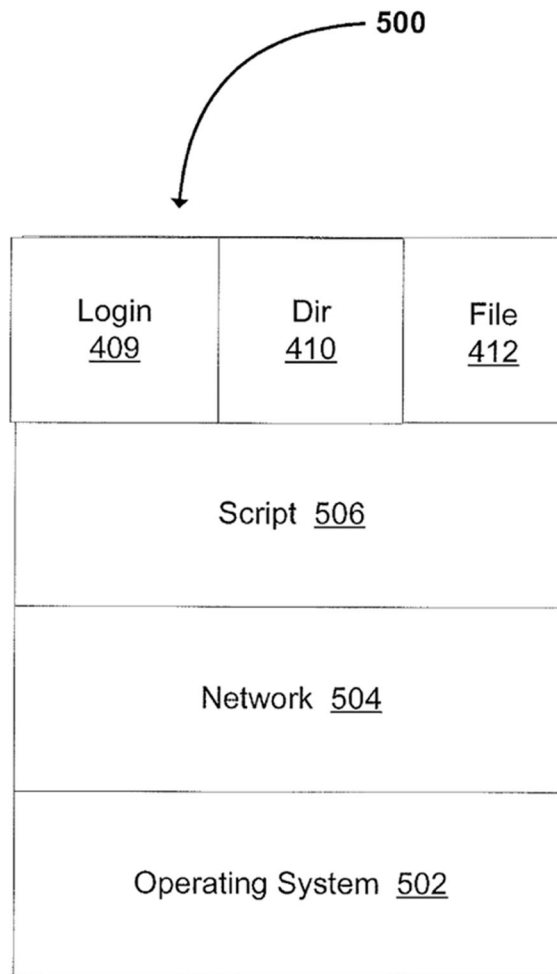


Figure 5

*Id.*, 12:64–13:5, Fig. 5.

Murphy's "file administration program 412" can "create an index" of files contained in an "image file" that "may contain a listing of the relevant files, the size of the files, the file location and/or other information about the files." *Id.*, 14:50–60. Murphy explains that "the index file may be created from the image file, which may be stored on server 206," and that "a single image file may be created for one or more client computers 202." *Id.*, 14:60–15:2. Where a single image file is used for multiple client computers, it "may be downloaded from server 20[6] during the pre-boot process, or as necessary to aid in system configuration and maintenance." *Id.*, 15:2–5.

## **2. Motivation to Combine Menage and Murphy and Reasonable Expectation of Success**

It would have been obvious to a POSITA to incorporate the indexing techniques taught by Murphy with the Menage shared/private storage system. EX1003, ¶¶368–78. Specifically, it would have been obvious to a POSITA to implement the creation of an index of files by one of Menage's "virtual processes 101" for the set of shared storage units and storage of the index on the "storage device 204" for access by other virtual processes that share the same set of shared storage units. *Id.*, ¶372. It likewise would have been obvious to a POSITA to implement the creation of an index for the sets of private storage units as well, since the private storage units may contain additional files created by a virtual process that are not found in the set of shared storage units. *See id.*, ¶427. Adding indexing to

the Menage system—which was well-known by 2004—would have provided the known benefit of more efficient file search and location. *Id.*, ¶377.

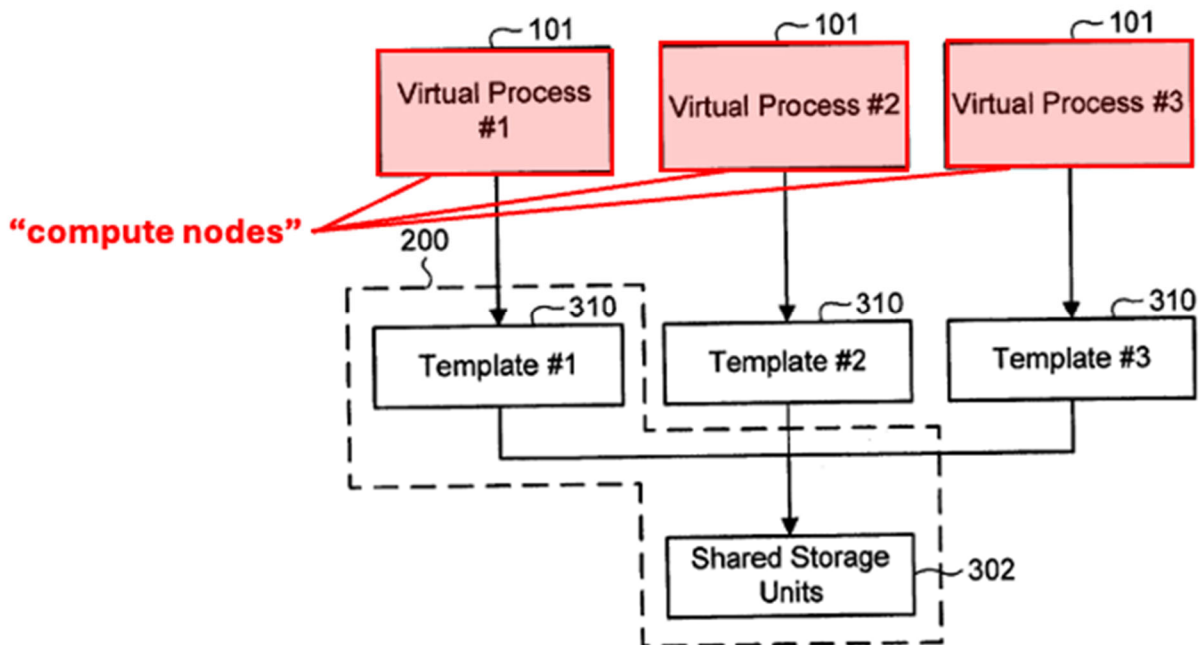
A POSITA would have had a reasonable expectation of success in adding indexing to the Menage system, in the same manner taught as Murphy, because file system indexing is a well-known computing technique, and both Menage and Murphy describe systems wherein multiple systems (*i.e.*, Menage’s “virtual processes” and Murphy’s “client computers”) share a base file system (*i.e.*, Menage’s “set of shared storage units” and Murphy’s “image file”). *Id.*, ¶378. The addition of Murphy’s teaching of indexing to the Menage system would have been nothing more than the use of a known technique (Murphy’s indexing) to improve similar devices (the Menage shared/private storage system) in the same way to yield predictable results (more efficient file searching and location through the creation of index files). *Id.*, ¶376. Thus, adding indexing to the Menage system would have been nothing more than a predictable use of a prior art element according to its established function. *KSR Int’l Corp. v. Teleflex, Inc.*, 550 U.S. 398, 417 (2007).

### 3. Detailed Analysis of Ground 2

#### a. Independent Claim 14

- 1) [14.pre] “A system for indexing file systems for a plurality of compute nodes, the system comprising:”

If claim 14’s preamble is limiting, the combination of Menage and Murphy discloses and/or renders it obvious. EX1003, ¶¶379–82. As explained for [1.pre], Menage “relates to file system creation ... particularly in the context of multiple virtual private servers running on a single physical host machine.” EX1004, 3:56–58. “[T]he file system 200 of each virtual process 101” (*i.e.*, “**a plurality of compute nodes**”) is “a combination of the set of shared storage units 302 and the set of private storage units 302 from the corresponding template 310”:



*Id.*, 8:63–67, Fig. 7.

Menage does not explicitly describe “indexing” of file systems; however, Murphy teaches this limitation. EX1003, ¶380. Murphy discloses a “file administration program 412” that can “create an *index*” of files that may “contain a listing of the relevant files, the size of the files, the file location and/or other information about the files.” EX1005, 14:52–58. Murphy explains that “the index file may be created from the image file, which may be stored on server 206,” and that “a single image file may be created for one or more client computers 202.” *Id.*, 14:58–15:2.

Accordingly, Menage and Murphy teach a “system for indexing file systems for a plurality of compute nodes.” EX1003, ¶382.

**2) [14.a] “a first storage unit configured to store blocks of a root image of said compute nodes;”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders it obvious as discussed for [1.a]. EX1003, ¶383.

**3) [14.b.i] “a plurality of second storage units configured to store leaf images of respective compute nodes,”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for [1.b.i]. EX1003, ¶384.

- 4) [14.b.ii] “said leaf images comprising only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of said root image, wherein said leaf images of respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes; and”

Menage/Murphy discloses and/or renders this limitation obvious because  
Menage discloses and/or renders this limitation obvious as discussed for [1.b.ii].  
EX1003, ¶385.

- 5) [14.c] “a plurality of union block devices corresponding to said compute nodes, said union block devices configured to interface between said compute nodes and said first and second storage units to distribute said file systems to said compute nodes, wherein said union block devices are configured to create said file systems by merging the blocks of said root image stored on the first storage unit with the blocks of respective leaf images stored on respective second storage units,”

Menage/Murphy discloses and/or renders this limitation obvious because  
Menage discloses and/or renders this limitation obvious as discussed for claim 4.  
EX1003, ¶386.

- 6) [14.d] “and wherein further at least one of said compute nodes is configured to index said root image and provide the indexing results to another of said compute nodes.”

Menage/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶387–95. Although Menage may not describe “indexing,” Murphy teaches the indexing of an “image file” (*i.e.*, a “root image”), as well as the ability to provide the indexing results to another client computer by storing an “index file” on a server. *Id.*, ¶¶388–90.

Murphy discloses a “file management program 412” that can “create an *index*” of files contained in an “image file” that “may contain a listing of the relevant files, the size of the files, the file location and/or other information about the files.” EX1005, 14:50–58. Murphy explains that “the index file may be created from the image file, which may be stored on server 206,” and that “a single image file may be created for one or more client computers 202.” Where a single image file is used for multiple client computers, it “may be downloaded from server 20[6] during the pre-boot process, or as necessary to aid in system configuration and maintenance.” *Id.*, 14:58–15:2.

It would have been obvious to a POSITA to incorporate indexing as taught by Murphy with the Menage system. Menage teaches the use of a “set of shared storage units 302” that is used to store data that is common to multiple virtual processes, and forms “an initial file system.” (*i.e.*, “a root image”). EX1004, 8:63-67. Menage’s

“shared storage units” are akin to Murphy’s description of a shared “single image file” that is “use for multiple client computers,” thus, it would have been obvious to create an index for the “shared storage units.” Furthermore, Menage’s “set of shared units 302” are located within a “storage device 204” that is accessible by all of the virtual processes that utilize the shared storage units, which is analogous to Murphy’s “server 206,” which stores the shared “image file.” Thus, it would have been obvious to a POSITA to store an index for the “set of shared storage units” on the “storage device 204,” so that the index can be provided (*i.e.*, accessed) by the virtual processes (*i.e.*, “provid[ing] the indexing results to another of said compute nodes”). Moreover, because Murphy’s indexing was a well-known technique for providing efficient file searching, adding it to the Menage system would have been nothing more than the use of a known technique to improve a similar device in a similar way. EX1003, ¶¶391–94. For the same reason, a POSITA would have had at least a reasonable expectation of successfully modifying Menage in view of Murphy in the manner described above.

Accordingly, the combination of Menage and Murphy teaches that “at least one of said compute nodes is configured to index said root image and provide the indexing results to another of said compute nodes.” *Id.*, ¶395.

**b. Claim 15**

**“The system as recited in claim 14 wherein said first storage unit and said second storage units are contained within a single storage appliance.”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for claim 3. EX1003, ¶396.

**c. Claim 16**

**“The system as recited in claim 14 further comprising: a plurality of union block devices configured to interface between respective compute nodes and said first storage unit and respective second storage units, said union block devices configured to distribute application environments to the compute nodes, wherein said union block devices are configured to create said application environments by merging the blocks of said root image with the blocks of respective leaf images.”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as explained above for claim 4. EX1003, ¶397.

**d. Claim 17**

**“The system as recited in claim 16 wherein said union block devices comprise low-level drivers for interfacing between the file systems of respective compute nodes and said first storage unit, respective second storage units, and said cache.”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for claim 5. EX1003, ¶398.

**e. Claim 18**

**“The system as recited in claim 14 wherein said first storage unit is read-only.”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for claim 6. EX1003, ¶399.

**f. Independent Claim 19**

**1) [19.pre] “A method for indexing file systems for a plurality of compute nodes, comprising:”**

If claim 19’s preamble is limiting, the combination of Menage and Murphy discloses and/or renders it obvious as discussed for [14.pre]. EX1003, ¶400.

**2) [19.a] “storing blocks of a root image of said compute nodes on a first storage unit;”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for [14.a]. EX1003, ¶401.

**3) [19.b.i] “storing leaf images for respective compute nodes on respective second storage units,”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for [14.b.i]. EX1003, ¶402.

**4) [19.b.ii] “said leaf images comprising only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of the root image, wherein said leaf images for respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes;”**

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for [14.b.ii]. EX1003, ¶403.

- 5) [19.c] “merging the blocks of said root image with the blocks of respective leaf images stored on respective second storage units to create respective file systems for respective compute nodes;”

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for [14.c]. EX1003, ¶404.

- 6) [19.d] “receiving indexing results pertaining to said root image from one of said compute nodes; and providing said indexing results to the others of said compute nodes.”

Menage/Murphy discloses and/or renders this limitation obvious because Menage discloses and/or renders this limitation obvious as discussed for [14.d]. EX1003, ¶405.

**g. Claim 20**

**“The method as recited in claim 19 further comprising:  
storing said indexing results on a shared storage unit.”**

Menage/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶406–11. Murphy discloses that an “index file” (*i.e.*, “said indexing results”) “may be stored on server 206” (*i.e.*, “on a shared storage unit”). *See* [14.d]; EX1005, 14:52–62. It would have been obvious to a POSITA that an “index file” as taught by Murphy that corresponds to the shared storage units 302 would have been stored

within Menage's storage device 204, which in one embodiment stores the shared storage units, the private storage units, as well as the usage maps. EX1003, ¶409.

Accordingly, Menage and Murphy teach claim 20. *Id.*, ¶411.

**h. Claim 21**

**“The method as recited in claim 19 wherein said merging occurs at an operational level between respective file systems of the compute nodes and said first storage unit and respective second storage units.”**

Menage discloses and/or renders this limitation obvious as discussed for claim 5. EX1003, ¶412.

**i. Claim 22**

**“The method as recited in claim 19 wherein said first storage unit is read-only.”**

Menage discloses and/or renders this limitation obvious as discussed for claim 6. EX1003, ¶413.

**j. Independent Claim 23**

**1) [23.pre] “A computer-readable storage medium having instructions stored thereon that, in response to execution by at least one computing device, cause the at least one computing device to:”**

If claim 23's preamble is limiting, Menage/Murphy discloses and/or renders it obvious as discussed for [1.pre] based on Murphy's teachings. EX1003, ¶414.

- 2) **[23.a] “receive data blocks of a file system, said data blocks comprising a root image portion and leaf image portion,”**

Menage/Murphy discloses and/or renders this limitation obvious as discussed for [1.a] and [1.b.i] because Menage discloses and/or renders these limitations obvious. EX1003, ¶415.

- 3) **[23.b] “said leaf image portion comprising only additional data blocks not previously contained in said root image portion and, changes made by said first compute node to the blocks of said root image, wherein said leaf image portion does not include blocks of said root image that are unchanged by said first compute node,”**

Menage/Murphy discloses and/or renders this limitation obvious as discussed for [1.b.ii] because Menage discloses and/or renders this limitation obvious. EX1003, ¶416.

- 4) **[23.c] “wherein said file system is the result of merging said root image portion and said leaf image portion together at the block-level;”**

Menage/Murphy discloses and/or renders this limitation obvious as discussed for claim 4 because Menage discloses and/or renders this limitation obvious. EX1003, ¶417.

**5) [23.d] “index said root image portion; and provide the results of said indexing to a second compute node,”**

Menage/Murphy discloses and/or renders this limitation obvious as discussed for [14.d] because Menage discloses and/or renders this limitation obvious. EX1003, ¶418.

**6) [23.e] “wherein said logic3 encoded in the one or more tangible media comprise computer executable instructions executed by the first compute node.”**

Menage/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶419–23. Menage’s system is described “in the context of multiple virtual private servers,” (*i.e.*, “compute nodes”) which it also refers to as “virtual processes,” that are “running on a single physical host machine.” EX1004, 3:56–58. Thus, it would have been obvious to a POSITA that this limitation merely describes the general operation of a “virtual private server,” which executes instructions. EX1003, ¶421.

---

<sup>3</sup> “[S]aid logic encoded in the one or more tangible media” has no antecedent basis in claim 23. For purposes of this Petition only, Petitioner interprets this phrase as referring to the “instructions” recited in [23.pre].

**k. Claim 24**

**“The computer-readable storage medium of claim 23, wherein the instructions further cause the at least one computing device to store said results of said indexing on a shared storage unit accessible by said second compute node.”**

Menage/Murphy discloses and/or renders this limitation obvious as discussed for claim 20 because Menage discloses and/or renders it obvious. EX1003, ¶424.

**l. Claim 25**

**“The computer-readable storage medium of claim 23, wherein the instructions further cause the at least one computing device to index said leaf image portion.”**

Menage/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶425–29. It would have been obvious to a POSITA to incorporate indexing of the “shared storage units” (*i.e.*, the “root image portion”) in the Menage system, as taught by Murphy. It would have been likewise obvious to a POSITA to incorporate indexing of the individual sets of “private storage units” (*i.e.*, the “leaf image portion[s]”) such that any changes made by a virtual process to the “initial file system” (including the addition of new files) in the set of shared storage units would be indexed as well. *Id.*, ¶427.

Accordingly, Menage/ Murphy teaches that “the instructions further cause the at least one computing device to index said leaf image portion.” EX1003, ¶429.

**m. Claim 26**

**“The computer-readable storage medium of claim 25, wherein the instructions further cause the at least one computing device to re-index said file system by re-indexing said leaf image portion and merging the results of said re-indexing of said leaf image portion with said results of said indexing of said root image portion.”**

Menage/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶430–33. It would have been obvious to a POSITA to “re-index” the sets of “private storage units” (*i.e.*, the “leaf image portion[s]”) periodically as they are modified by their respective virtual processes to ensure that the index is up-to-date and includes all modifications and newly created files. Further, because the file system for each virtual process in the Menage system is the combination of the “initial file system” in the sets of shared storage units (*i.e.*, the “root image portion”) and the changes made to the initial file system that are stored in the set of private storage units, it would have been obvious to a POSITA that the indexes (including any re-index) of the set of private storage units and the set of shared storage units would need to be combined (or “merged”) to provide an index for the full set of data. *Id.*, ¶431.

Accordingly, Menage/Murphy teaches that “the instructions further cause the at least one computing device to re-index said file system by re-indexing said leaf

image portion and merging the results of said re-indexing of said leaf image portion with said results of said indexing of said root image portion.” *Id.*, ¶433.

**n. Claim 27**

**“The computer-readable storage medium of claim 23 wherein said root image portion is read-only.”**

Menage/Murphy discloses and/or renders this limitation obvious as discussed for claim 6 because Menage discloses and/or renders this limitation obvious. EX1003, ¶434.

**C. Ground 3: Obviousness over Birse in view of Rothman (Claims 1–13)**

**1. Summary of Birse**

Birse relates to “supplying a reliable and maintainable operating system in a net-booted environment.” EX1006, Abstract. Specifically, Birse describes a “network computer system” or “NC system” that includes “one or more NC clients and one or more NC servers.” *Id.*, 4:41–43. An “NC client” refers to a client “that boots by accessing a copy of the operating system over a network,” and an “NC server” refers to a computer system “that services requests of NC clients,” such as by “provid[ing] a copy of the operating system to an NC client in response to a boot request.” *Id.*, 4:31–40.

**2. Summary of Rothman**

Rothman relates to techniques for “improving network boot efficiency.” Specifically, Rothman describes a system where “boot images that are initially sent

from a boot server to various clients are cached at network devices along communication paths between the boot server and the clients.” Upon subsequent requests for the boot images by clients, “boot images cached at the network devices are downloaded directly from the network devices rather than from the boot server, reducing network traffic.” EX1007, Abstract.

### **3. Motivation to Combine Birse and Rothman and Reasonable Expectation of Success**

It would have been obvious to a POSITA to incorporate the caching techniques taught by Rothman with the Birse network computer system. EX1003, ¶¶435–46. Specifically, it would have been obvious to a POSITA to cache blocks of the “system volumes 160” at the “NC server 170” such that, upon subsequent requests for the cached blocks by an NC client, the blocks would be retrieved from the cache rather than the underlying system volume. *Id.*, ¶¶439, 444. Adding caching to the Birse system—which was in widespread use long before 2004—would have provided the benefit of faster access to frequently requested data. *Id.*, ¶445.

A POSITA would have had a reasonable expectation of success in adding caching to Birse, in the same manner taught as Rothman, because caching is an ubiquitous (and decades-old) technique used in computer storage. Moreover, both Birse and Rothman describe systems wherein clients are booted by accessing a boot image over a network. *Id.*, ¶¶436–39. The addition of Rothman’s teaching of

caching to the Birse system would have been nothing more than the use of a known technique (Rothman's caching) to improve similar devices (the Birse NC system) in the same way to yield predictable results (faster access to frequently requested data). *Id.*, ¶444. Thus, adding caching to the Birse system would have been nothing more than a predictable use of a prior art element according to its established function. *KSR*, 550 U.S. at 417.

#### **4. Detailed Analysis of Ground 3**

##### **a. Independent Claim 1**

###### **1) [1.pre]**

If claim 1's preamble is limiting, Birse/Rothman disclose its and/or renders it obvious. EX1003, ¶¶447–49. Birse relates to an “NC system” that includes “one or more NC clients,” (*i.e.*, “**a plurality of compute nodes**”), 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”:

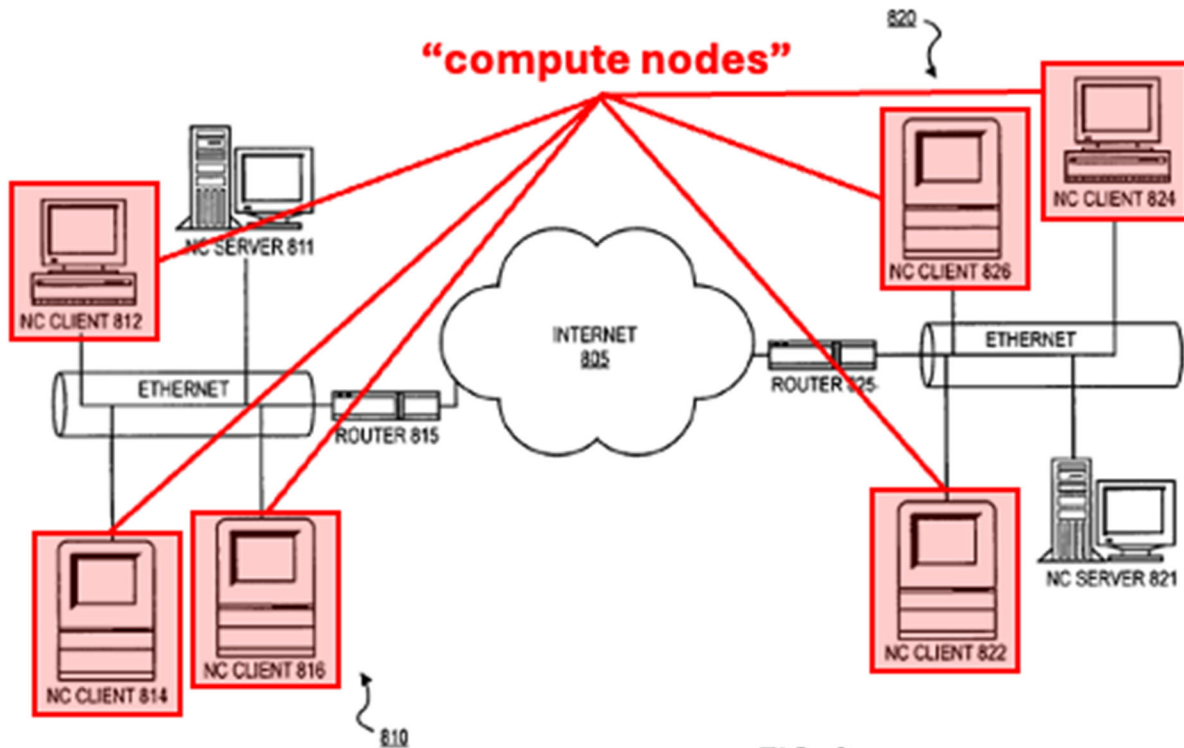


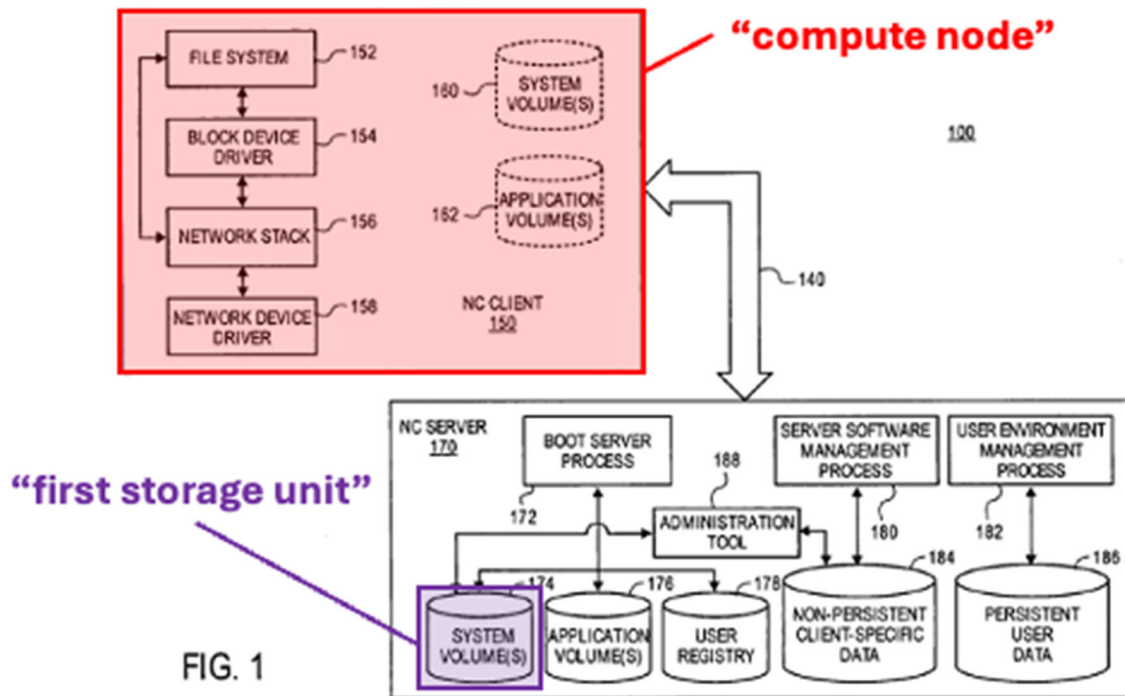
FIG. 8

EX1006, 4:31–40, Fig. 8.

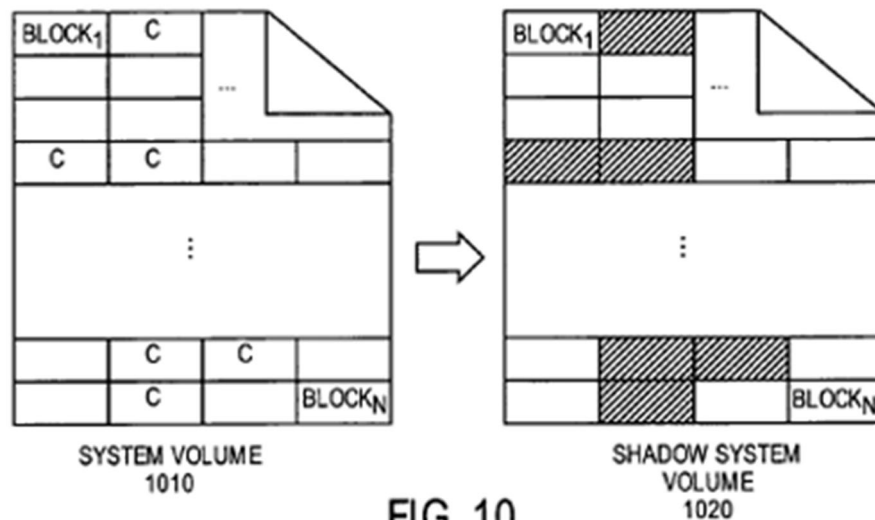
Accordingly, Birse/Rothman teaches [1.pre]. EX1003, ¶449.

2) [1.a]

Birse/Rothman discloses and/or renders this limitation obvious. EX1003, ¶¶450–54. Birse’s “system volumes 174” are “a first storage unit configured to store blocks of a root image of said compute nodes” as required by this limitation. *Id.*, ¶450. The “NC server 170 includes one or more system volumes 174,” (*i.e.*, “**a first storage unit**”), that “include a protected read-only, master copy of the operating system software”:



EX1006, 5:10–14, Fig. 1. Furthermore, Birse depicts a system volume that is configured to store blocks:



*Id.*, Fig. 10. Birse also explains that the NC client includes “a block device driver 154” that “services file system read and write requests to the system volumes 160.” *Id.*, 7:6–36.

Accordingly, Birse/Rothman teaches [1.a]. EX1003, ¶454.

**3) [1.b.i]**

Birse/Rothman discloses and/or renders this limitation obvious. EX1003, ¶¶455–57. Birse’s client-specific “shadow volumes,” which are “for the purpose of storing user preferences and/or changes to the operating system” are “a plurality of second storage units configured to store leaf images of respective compute nodes,” as required by this limitation. The “shadow volumes” associated with each NC client (*i.e.*, “**a plurality of second storage units**”) “contain[] 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” (*i.e.*, “leaf images of respective compute nodes”):

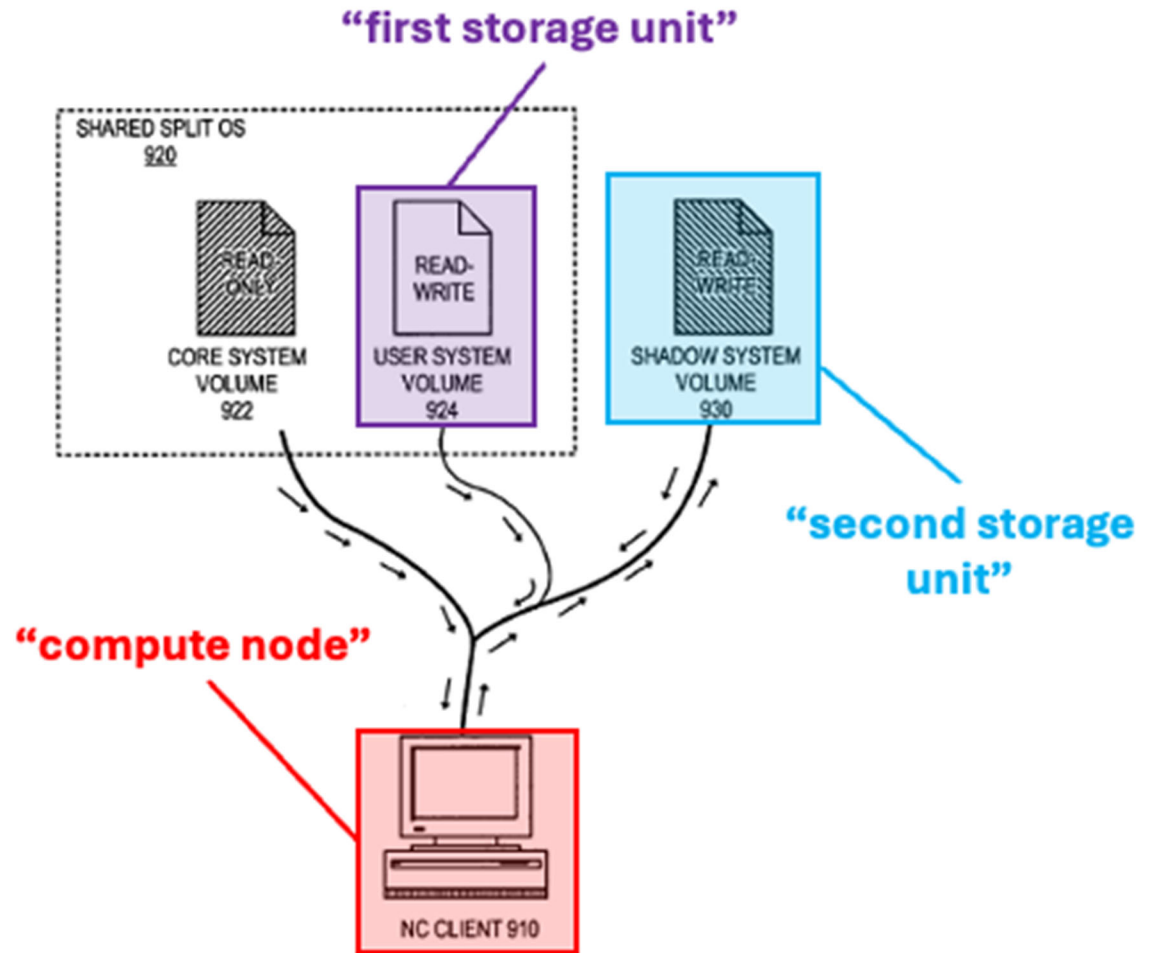


FIG. 9

*Id.*, 11:49–52, Fig. 9. The “shadow system volume 930 shadows the user system volume 924 by storing modifications that are made to the user system volume 924,” and “the NC server 170 may provide a separate user system volume 924 for each connected NC client.” *Id.*, 14:49–55.

Accordingly, Birse/Rothman teaches [1.b.i]. EX1003, ¶457.

**4) [1.b.ii]**

Birse/Rothman discloses and/or renders this limitation obvious. EX1003, ¶¶458–66. Birse explains that “one or more shadow volumes” (*i.e.*, the “leaf images”) may be created for the NC client ... for the purpose of storing user preferences *and/or changes to the operating system*,” which is stored on one or more “system volumes” (*i.e.*, the “root image”). EX 1006, 7:42–46. Moreover, “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.” *Id.*, 11:49–52.

“[A]t step 410, the file system 152 generates a write request directed to the system volumes 160,” which “is received by the block device driver 154” at step 420:

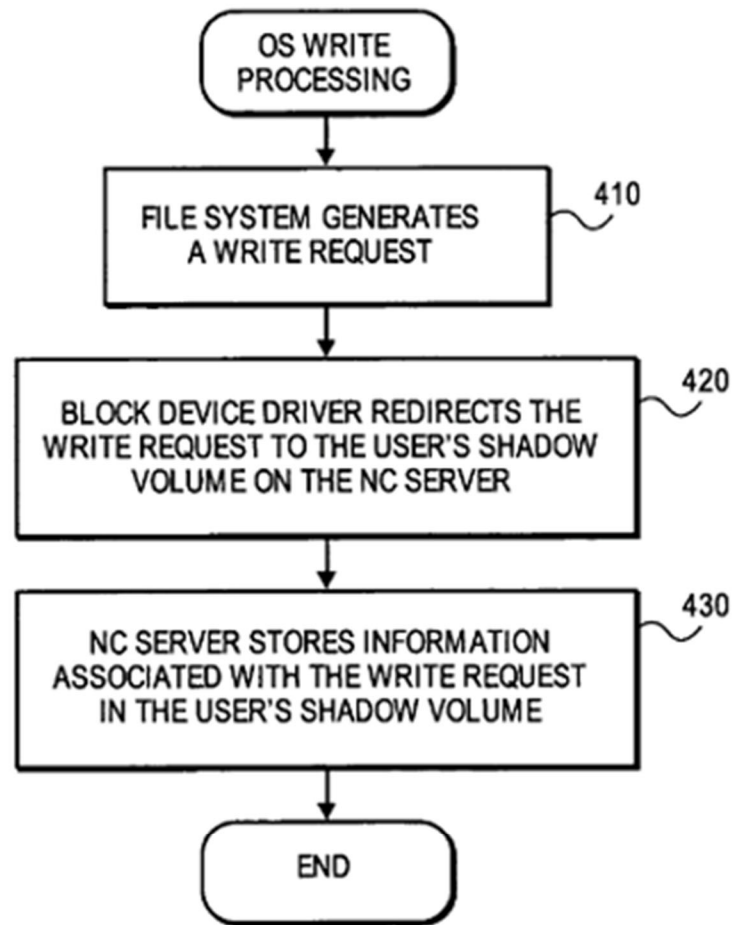


**FIG. 4**

*Id.*, Fig. 4. Rather than writing to the system volume, the write request is “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.” *Id.*, 7:6–11. Then, at step 430, “the NC server 170 stores information associated with the write request in the shadow volume associated with the NC client 150.” *Id.*, 7:12–17.

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:

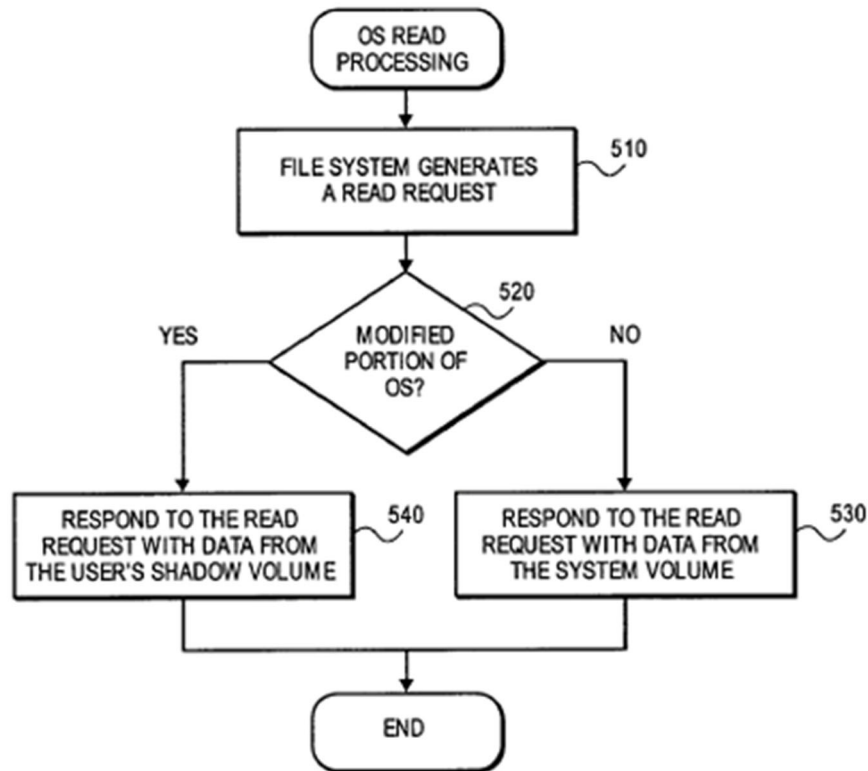


**FIG. 5**

*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.

The NC server “creates a shadow system volume 930 for each connected NC client” that “shadows the user system volume 924 by storing modifications that are made to the user system volume”:

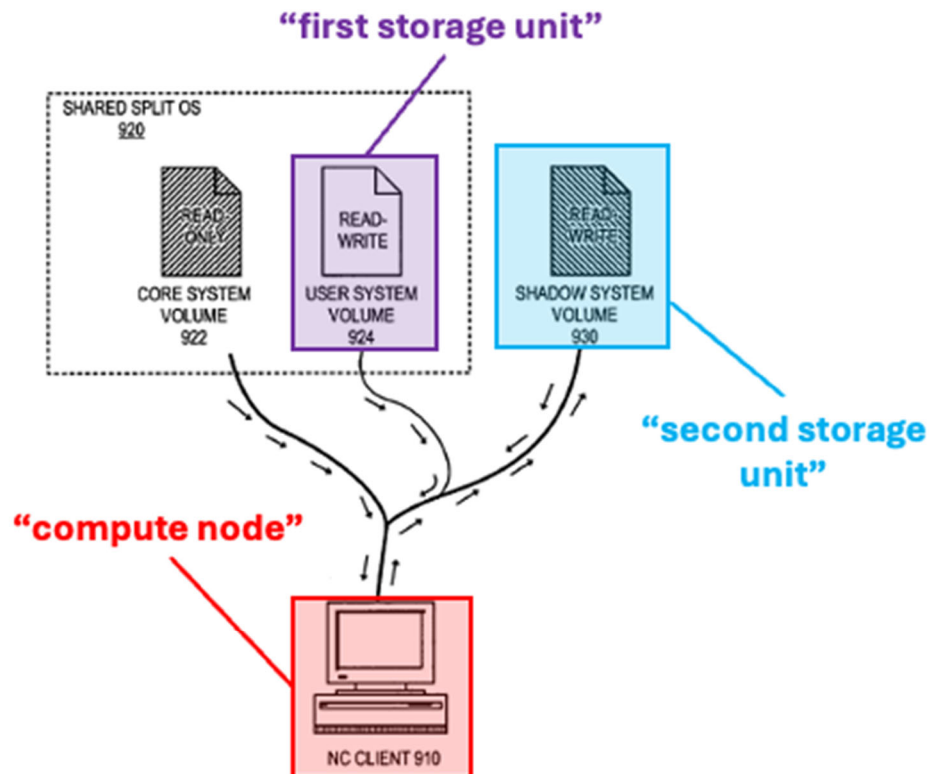


FIG. 9

*Id.*, 14:49–55, Fig. 9. The “shadow system volume[s]” are used by the block device driver 154 of each NC client “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,” which is “not visible to the user.” *Id.*, 14:56–62. As a result, “the user system volume 924 will appear to be the location where all writes are directed.” *Id.*, 14:63–64. As shown by the directional arrows in Figure 9, “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.” *Id.*, 15:1–7.

Accordingly, Birse/Rothman teaches [1.b.ii]. EX1003, ¶466.

**5) [1.c]**

Birse/Rothman discloses and/or renders this limitation obvious. EX1003, ¶¶467–75. Although Birse indicates that an NC client or an NC server may include “cache” memory and a “cache bus,” Birse does not specify how its cache is used. However, Rothman describes a system wherein “boot image file(s)” and “boot program bootstrap files” (*i.e.*, “blocks of said root image”) are cached at a network device or gateway server proximate to a client that is configured to boot using a “network boot process.” *Id.*, ¶¶468–69.

In one example of Rothman’s caching techniques, a client configured to boot using a network boot process “obtains an operating system boot image over the

network from a boot server.” EX1007, 4:6–7. “In conjunction with delivering the boot image to the client, the boot image *is cached at a network device ... or a gateway server* that is proximate to the client.” *Id.*, 4:7–12. Upon a subsequent system power on or reset of the client, “the client obtains the OS boot image from the caching device ... rather than the boot server.”

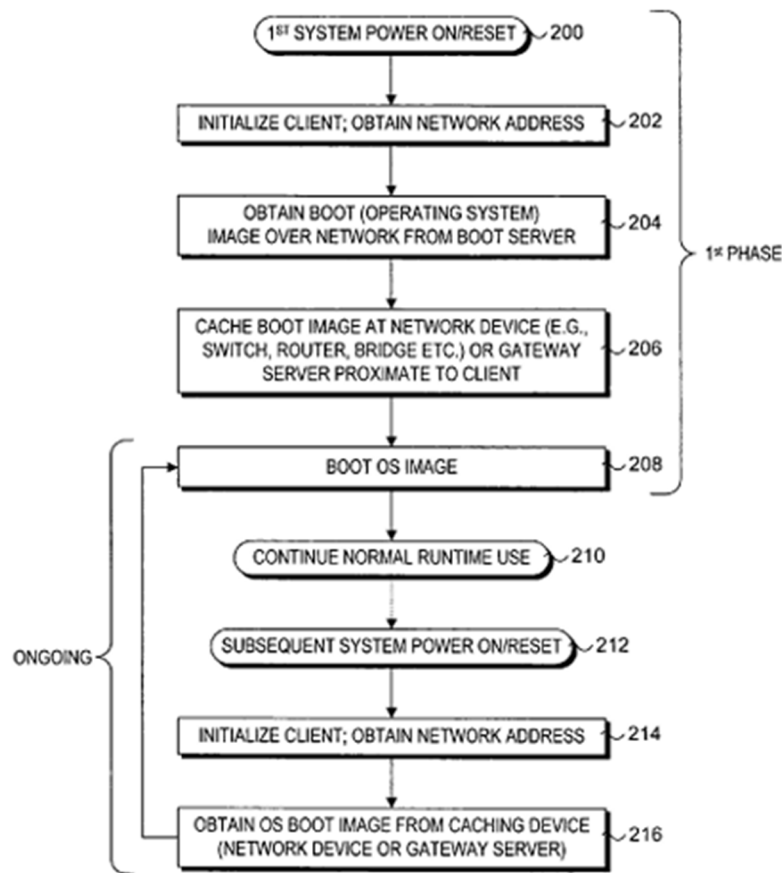


**Fig. 2**

*Id.*, 4:13–22, Fig. 2.

It would have been obvious to a POSITA to incorporate caching as taught by Rothman in Birse’s system to improve the performance of read requests directed to

the system volumes. Caching is a well-known and widely used technique. In a modified Birse system, cache memory would have been used to cache blocks of data from the system volumes upon access by one of the NC clients. On a subsequent read request for the same data (from any NC client), the data would have been retrieved by the block device driver from the faster cache memory rather than the system volume. EX1003, ¶473.

Accordingly, Birse/Rothman teaches [1.c]. *Id.*, ¶475.

**b. Claim 2**

Birse/Rothman, in light of the knowledge of a POSITA, disclose and/or render this limitation obvious for the same reasons that Menage, in light of the knowledge of a POSITA, renders claim 2 obvious. EX1003, ¶¶476–80; *see* [2] in Ground 1.

**c. Claim 3**

Birse/Rothman discloses and/or renders this limitation obvious. EX1003, ¶¶481–85. Birse contemplates the inclusion of a “cache” in an NC server. *See* [1.c]; EX1006, 7:63-8:1, 8:17–20. Likewise, Birse describes embodiments wherein the system volumes (*i.e.*, “first storage unit[s]”) and the shadow volumes (*i.e.* “said second storage units”) are also stored at an NC server. *Id.*, 12:30–63, Fig. 6.

Thus, Birse/Rothman teaches claim 3. EX1003, ¶485.

**d. Claim 4**

Birse/Rothman discloses and/or renders these limitations obvious. EX1003, ¶¶486–91. Birse teaches the use of a block device driver that redirects read and write

requests to/from a user's shadow volume as necessary. *See* [1.b.ii]. Accordingly, it would have been obvious to a POSITA that the block device drivers for each NC client are "a plurality of union block devices configured to interface between respective compute nodes and said first storage unit, respective second storage units, and said cache" as required by this claim." *Id.*, ¶487.

Furthermore, the block device driver, system volumes, and shadow volumes "distribute application environments to the compute nodes" as required by this claim. EX1003, ¶489. The '844 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." EX1001, 2:13–17. Additionally, "[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." *Id.*, 2:17–20.

The Birse system (including a cache as taught by Rothman) 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." EX1006, 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. Accordingly, 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. EX1003, ¶¶487–89.

Thus, Birse/Rothman teaches claim 4. *Id.*, ¶491.

**e. Claim 5**

Birse/Rothman discloses and/or renders these limitations obvious. EX1003, ¶¶492–95. 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 read and write requests between the system volumes and shadow volumes as appropriate. *See* [1.b.ii] and claim 4; EX1006, 12:11–15. And, “the block device driver 154 appears to the file system 152 to be a standard hard drive device driver.” *Id.*, 7:35–38.

Accordingly, Birse/Rothman teaches claim 5. EX1003, ¶495.

**f. Claim 6**

Birse/Rothman discloses and/or renders this limitation obvious. EX1003, ¶¶496–98. 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.” *See* [1.b.ii]; EX1006, 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 claim 6. EX1003, ¶497.

Accordingly, Birse/Rothman teaches claim 6. *Id.*, ¶498.

**g. Independent Claim 7**

**1) [7.pre]**

If claim 7’s preamble is limiting, Birse/Rothman renders it obvious as discussed for [1.pre]. EX1003, ¶499.

**2) [7.a]**

Birse/Rothman discloses and/or renders this limitation obvious as discussed for [1.a]. EX1003, ¶500.

**3) [7.b.i]**

Birse/Rothman discloses and/or renders this limitation obvious as discussed for [1.b.i]. EX1003, ¶501.

**4) [7.b.ii]**

Birse/Rothman discloses and/or renders this limitation obvious as discussed for [1.b.ii]. EX1003, ¶502.

**5) [7.c]**

Birse/Rothman discloses and/or renders this limitation obvious as discussed for [1.c]. EX1003, ¶503.

**h. Claim 8**

Birse/Rothman renders these limitations obvious. EX1003, ¶¶504–09. It would have been obvious to a POSITA that these limitations describe nothing more than the standard operation of a cache memory. *Id.*, ¶507. Rothman explains that “the boot image file(s) is/are *cached* at the network device or gateway server *proximate to the client*,” which results in faster retrieval than from the “boot server,” which may be located far away from the client. EX1007, 8:56–58; EX1003, ¶505.

In a modified Birse system, cache memory would have been used to cache blocks of data from the system volumes at the NC server upon access by one of the NC clients. On a subsequent read request for the same data (from any NC client), the data would have been retrieved by the block device driver from the faster cache memory rather than the system volume. EX1003, ¶¶506–07.

Accordingly, Birse/Rothman teaches claim 8. *Id.*, ¶508.

**i. Claim 9**

Birse/Rothman renders these limitations obvious. EX1003, ¶¶510–14. Like claim 8, it would have been obvious to a POSITA that this limitation describes nothing more than the standard operation of a cache. *Id.*, ¶512. As discussed for claim 9 in Ground 1, in the event of a cache miss, the data item is retrieved from the slower, underlying storage medium, and added to the cache memory for faster retrieval in response to a subsequent read request. Thus, it would have been obvious to a POSITA that any standard cache would have operated by “updating said cache memory based on said read request” when a data item is not stored in the cache memory. *Id.* Moreover, Rothman explains that “[i]n conjunction with delivering the boot image to the client” from the boot server upon an initial request (*i.e.*, the boot image is not in the cache), “*the boot image is cached* at a network device ... that is proximate to the client,” and that upon a subsequent boot, “the client obtains the OS image from the caching device ... rather than the boot server.” EX1007, 4:19–22, 8:56–58.

Accordingly, Birse/Rothman teaches claim 9. *Id.*, ¶514.

**j. Claim 10**

Birse/Rothman discloses and/or renders these elements obvious as discussed for claims 2 and 9. EX1003, ¶515.

**k. Claim 11**

Birse/Rothman discloses and/or renders this element obvious as discussed for [4.b]. EX1003, ¶516.

**l. Claim 12**

Birse/Rothman discloses and/or renders this element obvious as discussed for claim 5. EX1003, ¶517.

**m. Claim 13**

Birse/Rothman discloses and/or renders this element obvious as discussed for claim 6. EX1003, ¶518.

**D. Ground 4: Obviousness over Birse in view of Murphy (Claims 14–27)**

**1. Motivation to Combine Birse and Murphy and Reasonable Expectation of Success**

It would have been obvious to a POSITA to incorporate the indexing techniques taught by Murphy with the Birse network computing system. EX1003, ¶¶519–29. Specifically, it would have been obvious to a POSITA to implement the creation of an index of files by one of Birse’s “NC clients” for the “one or more system volumes 174” and storage of the index on the “NC server” for access by other NC clients. *Id.*, ¶542. It likewise would have been obvious to a POSITA to implement the creation of an index for the “shadow volumes” associated with each NC client as well, since the shadow volumes may contain additional files created by an NC client that are not found in the shared system volumes. *Id.*, ¶578. Adding

indexing to the Birse system—which was well-known by 2004—would have provided the known benefit of more efficient file search and location. *Id.*, ¶528.

A POSITA would have had a reasonable expectation of success in adding indexing to Birse, in the same manner taught as Murphy, because file system indexing is a well-known computing technique, and both Birse and Murphy describe systems wherein multiple systems (*i.e.*, Birse’s “NC clients” and Murphy’s “client computers”) share a base file system (*i.e.*, Birse’s “system volumes” and Murphy’s “image file”). *Id.*, ¶529. The addition of Murphy’s teaching of indexing to Birse would have been nothing more than the use of a known technique (Murphy’s indexing) to improve similar devices (the Birse network computing system) in the same way to yield predictable results (more efficient file searching and location through the creation of index files). *Id.*, ¶527. Thus, adding indexing to Birse would have been nothing more than a predictable use of a prior art element according to its established function. *KSR*, 550 U.S. at 417 (2007).

## **2. Detailed Analysis of Ground 4**

### **a. Independent Claim 14**

#### **1) [14.pre]**

If claim 14’s preamble is limiting, Birse/Murphy discloses and/or renders it obvious. EX1003, ¶¶530–34. Birse relates to an “NC system” that includes “one or more NC clients,” (*i.e.*, “**a plurality of compute nodes**”), 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”:

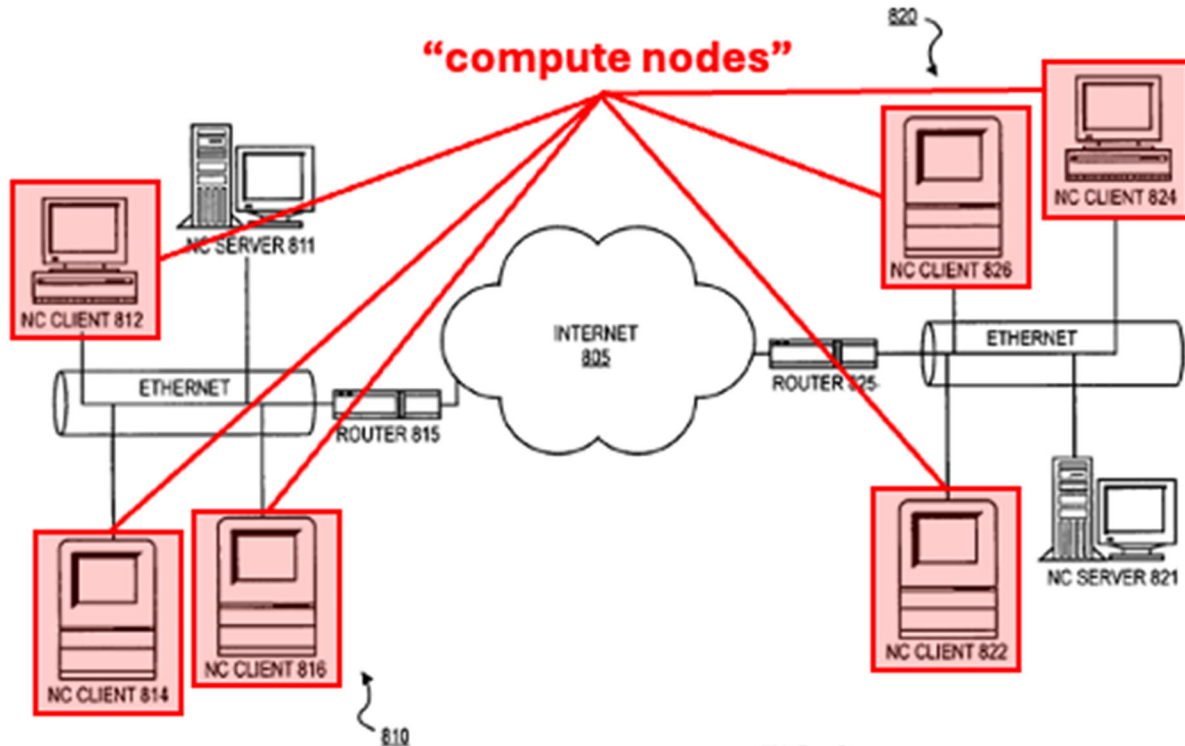


FIG. 8

See [1.pre]; EX1006, 4:31–44, Fig. 8.

Birse does not describe “indexing” of file systems; however, Murphy teaches this limitation. EX1003, ¶¶532–33. As explained in more detail below, Murphy discloses a “file administration program 412” that can “create an *index*” of files that “may contain a listing of the relevant files, the size of the files, the file location and/or other information about the files.” Murphy explains that “the index file may be created from the image file, which may be stored on server 206,” and that “a

single image file may be created for one or more client computers 202.” EX1005, 14:50–15:2.

Accordingly, Birse/Murphy teaches [14.pre]. EX1003, ¶534.

**2) [14.a]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [1.a] based on Birse’s teachings. EX1003, ¶535.

**3) [14.b.i]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [1.b.i] based on Birse’s teachings. EX1003, ¶536.

**4) [14.b.ii]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [1.b.ii] based on Birse’s teachings. EX1003, ¶537.

**5) [14.c]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 4 based on Birse’s teachings. EX1003, ¶538.

**6) [14.d]**

Birse/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶539–46. While Birse may not describe “indexing,” Murphy teaches indexing an “image file” (*i.e.*, a “root image”), as well as the ability to provide indexing results to another client computer by storing an “index file” on a server. *Id.*, ¶¶540–41.

It would have been obvious to a POSITA to incorporate indexing as taught by Murphy with the Birse system. Birse teaches the use of “one or more system volumes” that “include a protected read-only, master copy of the operating system software” (*i.e.*, “a root image”). EX1006, 5:9–14. Birse’s “system volumes” are akin to Murphy’s description of a shared “single image file” that is “use for multiple client computers,” thus, it would have been obvious to create an index for the “system volumes.” EX1003, ¶542. Furthermore, Birse’s “system volumes” are located within an “NC server” that is accessible by all of the NC clients that utilize the shared system volumes, which is analogous to Murphy’s “server 206,” which stores the shared “image file.” Thus, it would have been obvious to a POSITA to store an index for the “system volumes” on the NC server, so that the index can be provided (*i.e.*, accessed) by the NC clients (*i.e.*, “provid[ing] the indexing results to another of said compute nodes”). *Id.*, ¶543. Moreover, because Murphy’s indexing was a well-known technique for providing efficient file searching, adding it to the Birse network computing system would have been nothing more than the use of a known technique to improve a similar device in a similar way. *Id.*, ¶544. For the same reason, a POSITA would have had at least a reasonable expectation of successfully modifying Birse in view of Murphy in the manner described above.

Accordingly, Birse/Murphy teaches [14.d]. EX1003, ¶546.

**b. Claim 15**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 3 because Birse teaches this limitation. EX1003, ¶547.

**c. Claim 16**

Birse/Murphy discloses and/or renders these limitations obvious as discussed for claim 4 because Birse teaches this limitation. EX1003, ¶548.

**d. Claim 17**

Birse/Murphy discloses and/or renders these limitations obvious as discussed for claim 5 because Birse teaches this limitation. EX1003, ¶549.

**e. Claim 18**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 6 because Birse teaches this limitation. EX1003, ¶550.

**f. Independent Claim 19**

**1) [19.pre]**

If claim 19's preamble is limiting, Birse/Murphy discloses and/or renders it obvious as discussed for [14.pre]. EX1003, ¶551.

**2) [19.a]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [14.a]. EX1003, ¶552.

**3) [19.b.i]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [14.b.i]. EX1003, ¶553.

**4) [19.b.ii]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [14.b.ii]. EX1003, ¶554.

**5) [19.c]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [14.c]. EX1003, ¶555.

**6) [19.d]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [14.d]. EX1003, ¶556.

**g. Claim 20**

Birse/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶557–62. As explained regarding [14.d], Murphy discloses that an “index file” (*i.e.*, “said indexing results”) “may be stored on server 206” (*i.e.*, “on a shared storage unit”). EX1005, 14:52–15:5. It would have been obvious to a POSITA that an “index file” as taught by Murphy that corresponds to Birse’s system volumes 174 would have been stored within Birse’s NC server 170, which in one embodiment stores both the system volumes and the shadow volumes. EX1003, ¶560.

Accordingly, Birse/Murphy teaches claim 20. *Id.*, ¶562.

**h. Claim 21**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 5. EX1003, ¶563.

**i. Claim 22**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 6. EX1003, ¶564.

**j. Independent Claim 23**

**1) [23.pre]**

If claim 23's preamble is limiting, Birse/Murphy discloses and/or renders it obvious as discussed for [1.pre] regarding Birse. EX1003, ¶565.

**2) [23.a]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [1.a] and [1.b.i] regarding Birse. EX1003, ¶566.

**3) [23.b]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [1.b.ii] regarding Birse. EX1003, ¶567.

**4) [23.c]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 4 regarding Birse. EX1003, ¶568.

**5) [23.d]**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for [14.d] regarding Birse. EX1003, ¶569.

**6) [23.e]**

Birse/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶570–74. Birse describes a “network computer system” that includes one or more “NC clients” (*i.e.*, “compute nodes”) that utilize “machine-executable instructions.” EX1006, 7:6–20. Thus, it would have been obvious to a POSITA that this limitation merely describes the general operation of an “NC client,” which executes instructions. EX1003, ¶571.

**k. Claim 24**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 20. EX1003, ¶575.

**l. Claim 25**

Birse/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶576–80. It would have been obvious to a POSITA to incorporate indexing of the “system volumes 174” (*i.e.*, the “root image portion”) in the Birse system, as taught by Murphy. It would have been likewise obvious to a POSITA to incorporate indexing of the “shadow volumes” associated with each NC client (*i.e.*, the “leaf image portion[s]”) such that any changes made by an NC client to the operating

system (including the addition of new files) that is stored in the system volumes would be indexed as well. *Id.*, ¶¶577–78.

Accordingly, Birse/Murphy teaches claim 25. *Id.*, ¶580.

**m. Claim 26**

Birse/Murphy discloses and/or renders this limitation obvious. EX1003, ¶¶581–84. It would have been obvious to a POSITA to “re-index” the “shadow volumes” (*i.e.*, the “leaf image portion[s]”) periodically as they are modified by their respective NC clients to ensure that the index is up-to-date and includes all modifications and newly created files. *Id.*, ¶582. Further, because the file system for each NC client in the Birse system is the combination of the operating system software stored in the system volumes (*i.e.*, the “root image portion”) and the modifications to the original operating system that are stored in the shadow volumes, it would have been obvious to a POSITA that the indexes (including any re-index) of shadow volumes and the system volumes would need to be combined (or “merged”) to provide an index for the full set of data. *Id.*

Accordingly, Birse/Murphy teaches claim 26. *Id.*, 584.

**n. Claim 27**

Birse/Murphy discloses and/or renders this limitation obvious as discussed for claim 6 regarding Birse. EX1003, ¶585.

## **V. SECONDARY CONSIDERATIONS**

There are no secondary considerations of non-obviousness tied to the Challenged Claims. EX1003, ¶586. In any event, secondary considerations cannot overcome a strong prima facie case of obviousness. *Ohio Willow Wood Co. v. Alps South, LLC*, 735 F.3d 1333, 1344 (Fed. Cir. 2013).

## **VI. CONCLUSION**

Petitioner established a reasonable likelihood that the Challenged Claims are unpatentable. Petitioner therefore respectfully requests that IPR be instituted.

DATED: May 1, 2025

Respectfully Submitted,

/Brian M. Buroker/

Brian M. Buroker (Reg. No. 39,125)  
GIBSON, DUNN & CRUTCHER LLP  
1700 M St. NW  
Washington, DC 20036-4504  
Phone: (202) 955-8500  
Fax: (202) 467-0539  
Email: bburoker@gibsondunn.com

*Attorney for Petitioner Docker Inc.*

**CERTIFICATION UNDER 37 C.F.R. §42.24(d)**

Under the provisions of 37 C.F.R. §42.24(d), the undersigned attorney hereby certifies that the word count for Sections I–VII of the foregoing Petition for *Inter Partes* Review is 13,928, according to the word count tool in Microsoft Word.

DATED: May 1, 2025

By: /Brian M. Buroker/

Brian M. Buroker (Reg. No. 39,125)  
GIBSON, DUNN & CRUTCHER LLP  
1700 M St. NW  
Washington, DC 20036-4504  
Phone: (202) 955-8500  
Fax: (202) 467-0539  
Email: bburoker@gibsondunn.com

*Attorney for Petitioner Docker Inc.*

**CERTIFICATE OF SERVICE**

The undersigned certifies service pursuant to 37 C.F.R. §§42.6(e) and 42.105(a), (b) on the Patent Owner via FedEx overnight mail of a copy of this Petition for *Inter Partes* Review and supporting materials and exhibits, including a copy of Petitioner's Power of Attorney, at the Patent Owner at the correspondence address of record for the '844 patent:

3624 - VOLPE KOENIG  
30 SOUTH 17TH STREET, 18TH FLOOR  
PHILADELPHIA, PA  
UNITED STATES

DATED: May 1, 2025

By: /Brian M. Buroker/

Brian M. Buroker (Reg. No. 39,125)  
GIBSON, DUNN & CRUTCHER LLP  
1700 M St. NW  
Washington, DC 20036-4504  
Phone: (202) 955-8500  
Fax: (202) 467-0539  
Email: [bburoker@gibsondunn.com](mailto:bburoker@gibsondunn.com)

*Attorney for Petitioner Docker Inc.*

**APPENDIX: CHALLENGED CLAIM LISTING**

| <b>No.</b> | <b>Limitation</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [1.pre]    | A system for providing data to a plurality of compute nodes, the system comprising:                                                                                                                                                                                                                                                                                                                                                                                    |
| [1.a]      | a first storage unit configured to store blocks of a root image of said compute nodes;                                                                                                                                                                                                                                                                                                                                                                                 |
| [1.b.i]    | a plurality of second storage units configured to store leaf images of respective compute nodes,                                                                                                                                                                                                                                                                                                                                                                       |
| [1.b.ii]   | said leaf images including only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of said root image, wherein said leaf images of respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes; and                                                                                                                                    |
| [1c]       | a cache configured to cache blocks of said root image previously accessed by at least one of said compute nodes.                                                                                                                                                                                                                                                                                                                                                       |
| 2          | The system as recited in claim 1 wherein said cache is configured to store X most recently accessed blocks of said root image, and wherein X represents a cache threshold value.                                                                                                                                                                                                                                                                                       |
| 3          | The system as recited in claim 1 wherein said first storage unit, said second storage units, and said cache are contained within a single storage appliance.                                                                                                                                                                                                                                                                                                           |
| 4          | The system as recited in claim 1 further comprising: a plurality of union block devices configured to interface between respective compute nodes and said first storage unit, respective second storage units, and said cache to distribute application environments to the compute nodes, wherein said union block devices are configured to create said application environments by merging the blocks of said root image with the blocks of respective leaf images. |
| 5          | The system as recited in claim 4 wherein said union block devices comprise low-level drivers for interfacing between the file systems of respective compute nodes and said first storage unit, respective second storage units, and said cache.                                                                                                                                                                                                                        |
| 6          | The system as recited in claim 1 wherein said first storage unit is read-only.                                                                                                                                                                                                                                                                                                                                                                                         |

| No.      | Limitation                                                                                                                                                                                                                                                                                                                                              |
|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [7.pre]  | A method for providing data to a plurality of compute nodes, comprising:                                                                                                                                                                                                                                                                                |
| [7.a]    | storing blocks of a root image of said compute nodes on a first storage unit;                                                                                                                                                                                                                                                                           |
| [7.b.i]  | storing leaf images for respective compute nodes on respective second storage units,                                                                                                                                                                                                                                                                    |
| [7.b.ii] | said leaf images including only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of the root image, wherein said leaf images of respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes; and                      |
| [7.c]    | caching blocks of said root image that have been accessed by at least one of said compute nodes in a cache memory.                                                                                                                                                                                                                                      |
| 8        | The method as recited in claim 7 further comprising: receiving a read request from at least one of said compute nodes, wherein a first portion of the data requested is currently stored in said cache memory; and providing said first portion of said data to said at least one of said compute nodes from said cache memory.                         |
| 9        | The method as recited in claim 8 further comprising: updating said cache memory based on said read request.                                                                                                                                                                                                                                             |
| 10       | The method as recited in claim 9 wherein a second portion of the data requested is not currently stored in said cache memory and said updating comprises: caching said second portion in said cache memory; and removing the least recently accessed data from said cache memory if the amount of data in said cache memory is above a threshold value. |
| 11       | The method as recited in claim 7 further comprising: merging the blocks of said root image with the blocks of respective leaf images to create cohesive respective application environments.                                                                                                                                                            |
| 12       | The method as recited in claim 11 wherein said merging occurs at an operational level between file systems of the respective compute nodes and said first storage unit, respective second storage units, and said cache memory.                                                                                                                         |
| 13       | The method as recited in claim 7 wherein said first storage unit is read-only.                                                                                                                                                                                                                                                                          |

| No.       | Limitation                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [14.pre]  | A system for indexing file systems for a plurality of compute nodes, the system comprising:                                                                                                                                                                                                                                                                                                                                                                                                     |
| [14.a]    | a first storage unit configured to store blocks of a root image of said compute nodes;                                                                                                                                                                                                                                                                                                                                                                                                          |
| [14.b.i]  | a plurality of second storage units configured to store leaf images of respective compute nodes,                                                                                                                                                                                                                                                                                                                                                                                                |
| [14.b.ii] | said leaf images comprising only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of said root image, wherein said leaf images of respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes; and                                                                                                                                                            |
| [14.c]    | a plurality of union block devices corresponding to said compute nodes, said union block devices configured to interface between said compute nodes and said first and second storage units to distribute said file systems to said compute nodes,                                                                                                                                                                                                                                              |
| [14.d]    | wherein said union block devices are configured to create said file systems by merging the blocks of said root image stored on the first storage unit with the blocks of respective leaf images stored on respective second storage units,                                                                                                                                                                                                                                                      |
| [14.e]    | and wherein further at least one of said compute nodes is configured to index said root image and provide the indexing results to another of said compute nodes.                                                                                                                                                                                                                                                                                                                                |
| 15        | The system as recited in claim 14 wherein said first storage unit and said second storage units are contained within a single storage appliance.                                                                                                                                                                                                                                                                                                                                                |
| 16        | The system as recited in claim 14 further comprising: a plurality of union block devices configured to interface between respective compute nodes and said first storage unit and respective second storage units, said union block devices configured to distribute application environments to the compute nodes, wherein said union block devices are configured to create said application environments by merging the blocks of said root image with the blocks of respective leaf images. |

| No.       | Limitation                                                                                                                                                                                                                                                                                                                       |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 17        | The system as recited in claim 16 wherein said union block devices comprise low-level drivers for interfacing between the file systems of respective compute nodes and said first storage unit, respective second storage units, and said cache.                                                                                 |
| 18        | The system as recited in claim 14 wherein said first storage unit is read-only.                                                                                                                                                                                                                                                  |
| [19.pre]  | A method for indexing file systems for a plurality of compute nodes, comprising:                                                                                                                                                                                                                                                 |
| [19.a]    | storing blocks of a root image of said compute nodes on a first storage unit;                                                                                                                                                                                                                                                    |
| [19.b.i]  | storing leaf images for respective compute nodes on respective second storage units,                                                                                                                                                                                                                                             |
| [19.b.ii] | said leaf images comprising only additional data blocks not previously contained in said root image and changes made by respective compute nodes to the blocks of the root image, wherein said leaf images for respective compute nodes do not include blocks of said root image that are unchanged by respective compute nodes; |
| [19.c]    | merging the blocks of said root image with the blocks of respective leaf images stored on respective second storage units to create respective file systems for respective compute nodes;                                                                                                                                        |
| [19.d]    | receiving indexing results pertaining to said root image from one of said compute nodes; and providing said indexing results to the others of said compute nodes.                                                                                                                                                                |
| 20        | The method as recited in claim 19 further comprising: storing said indexing results on a shared storage unit.                                                                                                                                                                                                                    |
| 21        | The method as recited in claim 19 wherein said merging occurs at an operational level between respective file systems of the compute nodes and said first storage unit and respective second storage units.                                                                                                                      |
| 22        | The method as recited in claim 19 wherein said first storage unit is read-only.                                                                                                                                                                                                                                                  |
| [23.pre]  | A computer-readable storage medium having instructions stored thereon that, in response to execution by at least one computing device, cause the at least one computing device to:                                                                                                                                               |
| [23.a]    | receive data blocks of a file system, said data blocks comprising a root image portion and leaf image portion,                                                                                                                                                                                                                   |

| No.    | Limitation                                                                                                                                                                                                                                                                                                                           |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [23.b] | said leaf image portion comprising only additional data blocks not previously contained in said root image portion and, changes made by said first compute node to the blocks of said root image, wherein said leaf image portion does not include blocks of said root image that are unchanged by said first compute node,          |
| [23.c] | wherein said file system is the result of merging said root image portion and said leaf image portion together at the block-level;                                                                                                                                                                                                   |
| [23.d] | index said root image portion; and provide the results of said indexing to a second compute node,                                                                                                                                                                                                                                    |
| [23.e] | wherein said logic encoded in the one or more tangible media comprise computer executable instructions executed by the first compute node.                                                                                                                                                                                           |
| 24     | The computer-readable storage medium of claim 23, wherein the instructions further cause the at least one computing device to store said results of said indexing on a shared storage unit accessible by said second compute node.                                                                                                   |
| 25     | The computer-readable storage medium of claim 23, wherein the instructions further cause the at least one computing device to index said leaf image portion.                                                                                                                                                                         |
| 26     | The computer-readable storage medium of claim 25, wherein the instructions further cause the at least one computing device to re-index said file system by re-indexing said leaf image portion and merging the results of said re-indexing of said leaf image portion with said results of said indexing of said root image portion. |
| 27     | The computer-readable storage medium of claim 23 wherein said root image portion is read-only.                                                                                                                                                                                                                                       |