

(19) World Intellectual Property Organization
International Bureau



(43) International Publication Date
22 August 2002 (22.08.2002)

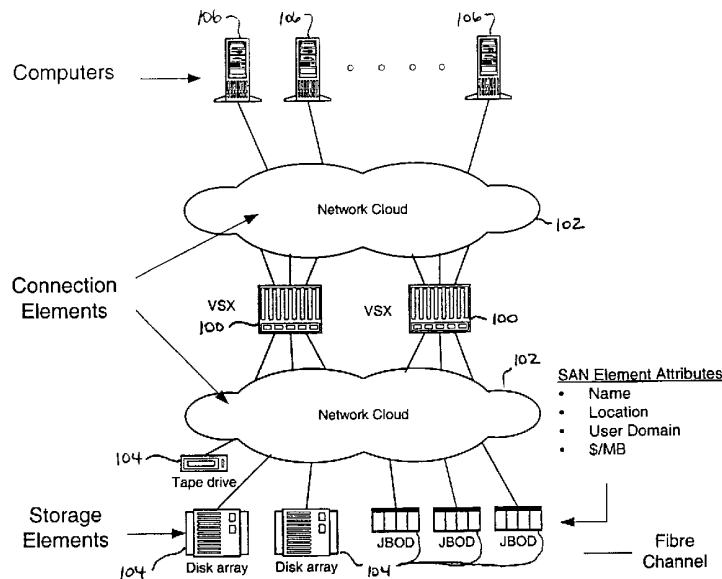
PCT

(10) International Publication Number
WO 02/065249 A2

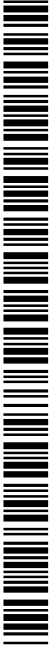
- (51) International Patent Classification⁷: **G06F** 69 Devonshire Way, San Francisco, CA 94131 (US). **PRASAD, Chandra** [IN/US]; 244 Merritt Rd., Los Altos, CA 94022 (US).
- (21) International Application Number: PCT/US02/04472
- (22) International Filing Date: 13 February 2002 (13.02.2002) (74) Agents: **HAMILTON, Charles, L.** et al.; TOWNSEND AND TOWNSEND AND CREW LLP, Two Embarcadero Center, 8th Floor, San Francisco, CA 94111 (US).
- (25) Filing Language: English
- (26) Publication Language: English (81) Designated States (*national*): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZM, ZW.
- (30) Priority Data: 60/268,694 13 February 2001 (13.02.2001) US (84) Designated States (*regional*): ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).
- (71) Applicant (*for all designated States except US*): **CONFLUENCE NETWORKS, INC.** [US/US]; 673 South Milpitas Blvd., Milpitas, CA 95035 (US).
- (72) Inventors; and
- (75) Inventors/Applicants (*for US only*): **GAJJAR, Kumar** [US/US]; 1700 Fan St., San Jose, CA 95131 (US). **COLLINS, Jim** [US/US]; 1 Royal Crest Dr., #10, Nashua, NH 03060 (US). **PATEL, Dipam** [IN/US]; 47679 Pima St., Fremont, CA 94539 (US). **MEYER, Richard** [US/US];

[Continued on next page]

(54) Title: STORAGE VIRTUALIZATION AND STORAGE MANAGEMENT TO PROVIDE HIGHER LEVEL STORAGE SERVICES



(57) Abstract: In a storage server on a storage area network, a media unit manager manages disk space in a manner transparent to the host computers. The media unit manager uses media units corresponding to the physical storage devices on the storage area network to build higher-level media units. The higher level media units can have the attributes of being sliced, concatenated, mirrored, striped, etc. Thus, the host computers need not be aware of the specific physical storage devices themselves, reducing the management and reconfiguration burden on the host computers when storage devices are added to or removed from the storage area network.



WO 02/065249 A2



Declaration under Rule 4.17:

— *of inventorship (Rule 4.17(iv)) for US only*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Published:

— *without international search report and to be republished upon receipt of that report*

STORAGE VIRTUALIZATION AND STORAGE MANAGEMENT TO PROVIDE HIGHER LEVEL STORAGE SERVICES

CROSS-REFERENCES TO RELATED APPLICATIONS

[01] This application claims priority to U.S. Provisional Application No. 60/268,694, filed February 13, 2001 and titled "Virtual Storage Systems", which is incorporated herein by reference.

STATEMENT AS TO RIGHTS TO INVENTIONS MADE UNDER FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

[02] NOT APPLICABLE

REFERENCE TO A "SEQUENCE LISTING," A TABLE, OR A COMPUTER PROGRAM LISTING APPENDIX SUBMITTED ON A COMPACT DISK.

[03] NOT APPLICABLE

BACKGROUND OF THE INVENTION

[04] The present invention relates to storage area networks and network attached storage. In particular, the present invention relates to a storage server that uses blocks of storage address space to virtualize different types of storage.

[05] Storage virtualization describes the process of representing, to a user, a number of discrete physical storage devices as a single storage pool having a single set of characteristics. For example, in a storage area network connecting host computers with storage devices, the user perceives a single block of disk space with a defined reliability (e.g., 100 GB at RAID1); however, the user's host computer is configured to access the storage devices such that 100 GB at RAID1 is provided, regardless of whether the data is stored on a single RAID1 disk array or is split across multiple, separate disks.

[06] In the above situation, each host computer must be aware of the storage devices connected to the storage area network because each host computer manages the storage

virtualization that is presented to its users. When the storage devices connected to the storage area network are modified (such as a new device being added or an existing device being removed), each host computer must be reconfigured to accommodate the modification. Such reconfiguration involves work by network administrators and ensures that changes in the network are not seamless.

[07] The present invention is directed toward improvements in this and other areas.

BRIEF SUMMARY OF THE INVENTION

[08] In order to provide improvements in storage area networking, the present invention uses “media units” to represent storage space and builds additional media units from existing media units. For example, the lowest level media units are physical media units. Higher-level media units can then be built from the physical media units. Examples of such higher-level media units include slice media units, concatenation media units, striped media units (also called RAID0 media units), and mirrored media units (also called RAID1 media units). Such higher-level media units can also be built from other higher-level media units. This greatly improves management of the storage area network from a host computer perspective.

[09] In accordance with one embodiment of the present invention, a storage server connects host computers and storage devices in a storage area network. The storage devices have associated therewith physical logical unit numbers (PLUNs). The storage server includes a processor that executes processing. The processing includes mapping the PLUNs to first media units; creating a second media unit from the first media units; and managing access, using the second media unit, to the storage devices by the host computers.

[10] The details of the embodiments of the present invention can be seen in the following drawings and detailed description.

BRIEF DESCRIPTION OF THE DRAWINGS

[11] FIG. 1 is a block diagram of a storage area network including a storage server according to an embodiment of the present invention;

[12] FIG. 2A is a block diagram of hardware components in the storage server according to an embodiment of the present invention;

- [13] FIG. 2B is a block diagram of management functions in the storage server according to an embodiment of the present invention;
- [14] FIG. 3A is a diagram of a native pool and a virtual pool according to an embodiment of the present invention;
- [15] FIG. 3B is a data diagram showing various types of media units according to an embodiment of the present invention;
- [16] FIGS. 4A-4B are diagrams of usage states and service states of media units according to an embodiment of the present invention;
- [17] FIGS. 5A-5B are diagrams showing distributed processing by the media unit manager according to an embodiment of the present invention;
- [18] FIG. 6 is a diagram showing the relationship between PLUNs, media units and VLUNs according to an embodiment of the present invention;
- [19] FIG. 7 is a diagram showing examples of usage states of media units according to an embodiment of the present invention;
- [20] FIG. 8 is a diagram of exemplary slice media units according to an embodiment of the present invention;
- [21] FIG. 9 is a diagram of an exemplary concatenation media unit according to an embodiment of the present invention;
- [22] FIG. 10 is a diagram of an exemplary mirrored media unit according to an embodiment of the present invention;
- [23] FIG. 11 is a diagram of exemplary mirrored and sliced media units according to an embodiment of the present invention;
- [24] FIG. 12 is a diagram of exemplary sliced and mirrored media units according to an embodiment of the present invention;
- [25] FIG. 13 is a diagram of an exemplary striped media unit according to an embodiment of the present invention;
- [26] FIG. 14 is a diagram of a reconstruction process using a mirrored media unit according to an embodiment of the present invention;
- [27] FIG. 15 is a diagram of a reconstruction process using slice media units and a mirrored media unit according to an embodiment of the present invention;

[28] FIG. 16 is a diagram of a snapshot process using a mirrored media unit according to an embodiment of the present invention;

[29] FIG. 17 is a diagram of a non-RAID to RAID conversion process according to an embodiment of the present invention;

[30] FIG. 18 is a diagram of a VLUN migration process according to an embodiment of the present invention;

[31] FIG. 19 is a diagram showing examples of media units with synchronous and asynchronous members according to an embodiment of the present invention;

[32] FIG. 20 is a diagram of another snapshot process according to an embodiment of the present invention;

[33] FIG. 21 is a diagram of yet another snapshot process according to an embodiment of the present invention; and

[34] FIG. 22 is a diagram of a process of merging media units according to an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[35] FIG. 1 shows a storage server 100 according to an embodiment of the present invention. The figure also shows a storage area network (SAN) 102, a number of physical storage devices 104, and a number of host computers 106. The storage server 100 is also referred to as a Virtual Storage Exchange (VSX) and is further detailed in FIGS. 2A-2B.

[36] The SAN 102 can be any type of computer network. It is referred to as a storage area network in the present application because that is its relevant function with respect to the embodiments of the present invention. In an embodiment of the present invention, the SAN 102 is a Fibre Channel network, the host computers 106 and the storage devices 104 are configured to communicate with a Fibre Channel network, and the storage server 100 is also configured to communicate with a Fibre Channel network. Thus, the storage server 100 can be easily added to an existing SAN.

[37] The physical storage devices 104 include tape drives, disk arrays, JBODs ("just a bunch of disks"), or other types of data storage devices. The physical storage devices 104 can be connected directly to the host computers 106 via the SAN 102 or can be indirectly connected to the host computers 106 via the SAN 102 and the storage server 100. As discussed above in the

Background, management of storage virtualization is burdensome when the storage devices 104 are directly connected to the host computers 106 via the SAN 102. The present invention improves management of storage virtualization by using the storage server 100 to indirectly connect the storage devices 104 to the host computers 106.

[38] The host computers 106 can be servers or stand-alone computers. The host computers 106 can be directly connected to the SAN 102 or indirectly connected via a switch, router, or other communication link.

[39] FIG. 2A is a block diagram of the storage server 100 showing the hardware components related to embodiments of the present invention, including a storage processor 110, a line card 112, a virtual server card 114, and a switch fabric 116.

[40] The storage server 100 may include one or more storage processors 110. The storage processors 110 process the storage commands and data to be stored as information flows between the host computers 106 and the storage devices 104. One or more of the storage processors 110 may be included on each line card 112. The storage server 100 includes space for numerous line cards 112, so the capabilities of the storage server 100 can be modularly increased by adding more line cards 112 or more storage processors 110. Each storage processor 110 is associated with one or more ports of the storage server 100.

[41] The storage server 100 may include one or more virtual server cards 114. The virtual server cards control the operation of the storage server 100 and control the line cards 112, which perform the actual work of transferring commands and data.

[42] The switch fabric 116 connects the storage processors 110. The switch fabric switches information received at one port to another port of the storage server 100. For example, when a host computer 106 wants to read data stored on the storage area network 102, its request is processed by the storage processor 110 associated with the port associated with that host computer 106. That storage processor 110 is referred to as the upstream storage processor 110. The upstream storage processor 110 communicates with a downstream storage processor 110 associated with the port associated with the storage device 104 storing the data to be read, via the switch fabric 116. Then the switch fabric 116 transfers the data read from the storage device to the host computer 106, via the downstream and upstream storage processors 110.

[43] FIG. 2B is a block diagram of the storage server 100 showing the functionality relevant to embodiments of the present invention. The functions of the storage server 100 may be

implemented by one or more processors that execute processing according to one or more computer programs, microcode segments, hardware structures, or combinations thereof. The functions relevant to the present invention are the media unit (MU) manager 120, the virtual logical unit number (virtual LUN or VLUN) manager 122, and the physical logical unit number (physical LUN or PLUN) manager 124. Additional details of the storage server 100 are provided in other applications assigned to the present assignee and filed on February 13, 2002 that claim the benefit from the above-noted Provisional Application No. 60/268,694 and are hereby incorporated herein by reference: Attorney Docket No. 20949P-000200US titled "Silicon Based Storage Virtualization Server", Attorney Docket No. 20949P-000500US titled "Method and Apparatus for Identifying Storage Devices", Attorney Docket No. 20949P-000600US titled "System and Method for Policy Based Storage Provisioning and Management", Attorney Docket No. 20949P-000700US titled "Virtual Data Center", Attorney Docket No. 20949P-000800US titled "Failover Processing in a Storage System", Attorney Docket No. 20949P-000900US titled "RAID at Wire Speed", and Attorney Docket No. 20949P-001000US titled "Method for Device Security in a Heterogeneous Storage Network Environment".

[44] The PLUN manager 124 manages data and command transfer to and from the storage devices 104. Each storage device 104 may have associated therewith a PLUN that is used for identifying each particular storage device 104.

[45] The VLUN manager 122 manages data and command transfer to and from the host computers 106. Each host computer 106 may be associated with one or more VLUNs. Each VLUN represents a virtual address space (e.g., gigabytes of storage) with defined attributes (e.g., performance parameters, reliability level, etc.). As such, each host computer 106 exchanges data and commands with the storage server 100 with reference to a particular VLUN.

[46] The MU manager 120 basically translates between VLUNs and PLUNs. The MU manager 120 is responsible for managing the address space of all the storage devices 104 (physical LUNs) connected to the storage server 100. The MU manager 120 also manages the address space of the storage constructs built within the storage server 100, including slices, concatenations, RAID0 (stripes) and RAID1 (mirrors).

[47] The MU manager 120 uses an abstract block-storage addressing technique that enables address spaces to be treated in a logical manner, regardless of the underlying storage constructs or physical LUNs. These logical address spaces can be combined together into more complex

and feature rich storage constructs, which are also treated simply as abstract block-storage address spaces.

[48] Used in conjunction with a virtual LUN, these logical address spaces can be configured to appear as LUNs on a multi-ported storage device. This process of presenting physical LUNs as logical address spaces on virtual devices is referred to as storage virtualization.

[49] Abstract block-storage addressing is achieved via a data structure known as a media unit (MU). The various types of media units are described throughout this document.

[50] FIG. 3A is a block diagram showing that the media unit manager 120, in one embodiment of the invention, maintains two major pools of media units 130: the native pool 132 and the virtual pool 134.

[51] The native pool 132 contains legacy direct access devices with existing user data and devices such as tape, CD-ROM, SES, etc. These media units 130 are preferably virtualized “as is” by the media unit manager 120. In other words, the media unit manager 120 preferably does not alter the content or the attributes of the underlying address space. A media unit 130 in the native pool 132 preferably represents an entire physical LUN.

[52] A newly discovered physical LUN is preferably added to the free list 136 of the native pool 132. A media unit 130 in the native pool 132 can be exported to a host using a VLUN. It preferably cannot be used as a building block for other media units 130. Direct access media units 130 in the native pool 132 can be migrated to the virtual pool 134.

[53] The native pool 132 preferably obeys the following rules:

[54] 1. A media unit 130 created for a newly discovered PLUN is preferably added to the native pool 132;

[55] 2. Media units 130 in the native pool are preferably virtualized “as is”; and

[56] 3. The native pool 132 preferably contains media units for all the devices that (a) are not direct access devices, such as tape, CD-ROM, SES, etc., and (2) are direct access devices that contain existing user data.

[57] Direct access media units 130 in the native pool 132 that are not currently mapped to a VLUN can be migrated to the free list 138 in the virtual pool 134. Thus, the virtual pool 132 preferably only contains media units 130 representing direct access devices. Once in the virtual pool 134, these media units 130 can be exported to a host 106 using a VLUN or they can be used as building blocks to create other types of media units.

[58] For example, the media unit manager 120 can use two media units in the virtual pool 134, each of logical address space of size X, to create a mirrored media unit (RAID1 media unit) with logical address space of size X.

[59] The Virtual Pool preferably obeys the following rules:

[60] 1. The maximum number of media units in the system is unlimited;

[61] 2. Media unit properties can be changed;

[62] 3. The starting and ending address of any media unit is preferably modulo 2 so that it can be calculated by shifting the address;

[63] 4. A media unit for a PLUN is migrated to the Virtual Pool preferably only if instructed by the administrator;

[64] 5. A media unit for a PLUN preferably can only be in one of the virtual pool 134 or the native pool 132, but not both; and

[65] 6. The virtual pool 134 preferably can only contain direct access devices.

[66] In one embodiment of the present invention, bookkeeping information may be written to media units in the virtual pool 134 to facilitate identification; for example, when a media unit is moved. If meta-data is maintained on the media unit, a newly-discovered media unit may be placed directly into the virtual pool 134.

[67] The media unit manager 120 uses media units 130 to represent the logical address space of a block storage device. Persistent information on media units 130 may be maintained within the database. In order to describe the functional characteristics of media units, the following attributes may be used:

[68] 1. Type: Identifies the kind of media unit and characteristics of the logical address space;

[69] 2. Usage State: Indicates the availability of the media unit;

[70] 3. Service State: Indicates the "online status" of the media unit;

[71] 4. Start LBA: The starting block of the logical address space (preferably 0);

[72] 5. Block Size: The size of a block in the logical address space; and

[73] 6. Block Count: The number of blocks in the logical address space.

[74] Media units may also be characterized by a set of properties, where a property is simply a name/value pair. These properties are based upon 1) the characteristics of the underlying PLUNs, 2) the properties of any child media units used as building blocks, and 3) the type of the

media unit itself. Media unit properties can be added, removed, and modified on the fly. Some example properties include:

- [75] 1. Performance: Fast, medium, slow, etc.;
- [76] 2. Connection Redundancy: Single path, multi-path, multi-storage server, etc.;
- [77] 3. Data Redundancy: Non-redundant, local redundancy, remote Redundancy, etc.;
- [78] 4. Location: An optional user-specified property; and
- [79] 5. Account: The account that owns the media unit -- may be aggregation or property.

[80] The media unit manager 120 provides a core set of interfaces to build the various types of media units 130 and to control the native pool 132 and the virtual pools 134, among other functions. Advanced, wizard-like interfaces to create media units that fulfill a set of quality of service requirements may be used.

[81] A typical user wishing to allocate a logical address space may simply specify one or more of these quality of service properties. The storage server 100 then queries the database for media units that meet those criteria. After some complex analysis, it creates the “best-fit” media unit using the core operations provided by the media unit manager 120. Advanced users that want complete control over the media units can bypass the wizard interfaces and access the operations of the media unit manager 120 directly.

[82] The native pool 132 contains legacy direct access devices with existing user data. It also contains many other devices that are preferably virtualized “as is” to a host using a VLUN. All media units 130 in the native pool 132 are preferably of the physical media unit type.

[83] A physical media unit is bound to a single physical LUN and preferably represents the entire address space of that physical LUN. The peripheral device type of the underlying PLUN indicates the type of the device, such as Direct Access, Sequential Access, Printer, Processor, Write Once, CD ROM, Scanner, Optical Memory, Medium Changer, Communications, ASCIT8, Storage Array Controller, Enclosure Services, Unknown, etc.

[84] Preferably, only those physical media units with a PLUN peripheral device type of “Direct Access” are migrated from the native pool 132 to the virtual pool 134.

[85] FIG. 3B is a data diagram showing the object model for the media units. The data diagram is depicted in the Unified Modeling Language, an industry standard way of representing object models. Information on the Unified Modeling Language can be found in the document “OMG Unified Modeling Language Specification” (Object Management Group), which is

incorporated herein by reference. Copies of this document may be obtained online at the OMG website <omg.org>.

[86] FIGS. 4A-4B show the usage states and service states of physical media units in the native pool 132 according to one embodiment of the present invention. Preferably all media units 130 in the native pool 132 share the same basic “pass-through” behavior from the point of view of the media unit manager 120. When a newly discovered PLUN is announced to the media unit manager 130, the media unit manager performs the following steps:

[87] 1. It creates a physical media unit and binds it to the underlying PLUN;

[88] 2. It sets the usage state to “Unused”;

[89] 3. It sets the service state to “Online”; and

[90] 4. It adds the physical media unit to the free list 136 in the native pool 132.

[91] When a physical media unit is exported to a host by mapping it to a VLUN using the virtual LUN manager 122, its usage state is set to “Used.” It will preferably remain in that state until the media unit is unmapped from the VLUN.

[92] If the physical LUN is deleted, the physical media unit (native or virtual, used or unused) will preferably have its state updated to Offline and Missing. The user can then choose to remove these media units manually. Should the PLUN be rediscovered on the same or a different port on the storage server 100, its corresponding physical media unit will preferably return to the “Online” state.

[93] The migration of a direct access physical media unit from the native pool 132 to the virtual pool 134 is preferably only allowed when the physical media unit (of direct access type) is in the “Unused” state.

[94] Similar diagrams may be used to illustrate common media unit usage states and service states in the virtual pool 134. The virtual pool 134 contains direct access media units 130 that can be (1) exported to a host by mapping the media unit to a VLUN, (2) used as building blocks to create other media units in the virtual pool, or (3) reserved for future use. The following types of media units 130 can be found in the virtual pool 134:

[95] 1. Physical media unit (whose PLUN peripheral device type is Direct Access);

[96] 2. Slice media unit;

[97] 3. Concatenation media unit;

[98] 4. Striped (RAID0) media unit; and

[99] 5. Mirrored (RAID1) media unit.

[100] The sections that follow describe the different types of media units found in the virtual pool 134 along with their state behavior and characteristics.

[101] The media units 130 in the virtual pool 134 share some common behavior. Each type of media unit then extends this behavior to achieve its specific responsibilities.

[102] If a media unit 130 is exported to a host by mapping it to a VLUN or it is used as a building block for another media units, its usage state is set to "Used." It will remain in that state until the media unit is no longer mapped to a VLUN and it is not used as a building block for other media units, at which time it re-enters the "Unused" state.

[103] A media unit 130 in the virtual pool 134 can enter the "Reserved" state to indicate that it is not available for general consumption as a building block, even though it has not yet been mapped to a VLUN. The "Reserved" state enables the user to create several media units in advance and map them to VLUNs at a later time.

[104] The "Reserved" state can also be used by the services software to temporarily remove a VLUN mapping without exposing the media unit to general consumption as a building block.

[105] FIGS. 5A-5B are block diagrams showing the interaction between software modules and storage processors 110 in the storage server 100 according to one embodiment of the present invention. The MU manager 120 uses distributed software agents, such as the data mover, to perform the work for processes such as RAID reconstruction, RAID initialization, snapshot, etc. (all of which are further described in subsequent sections). These software agents run on different storage processors 110 than the MU manager 120. This allows the MU manager 120 to scale as more ports and physical LUNs are added to the system.

[106] FIG. 6 shows the relationship between physical media units and the other services. The PLUN manager 124 manages PLUNs, the MU manager 120 manages media units, and the VLUN manager 122 manages VLUNs. These relationships are further developed in following sections.

[107] In addition, FIG. 6 shows the relationships between PLUNs, media units, and VLUNs. Generally, a PLUN directly corresponds to a storage device, such as a disk or a disk array. Such a direct one-to-one relationship generally shown in the following figures. However, a PLUN can also be associated with a portion of the storage device. Multiple PLUNs can be associated with different portions of a single storage device.

[108] Each physical media unit (first-level media unit) generally directly corresponds to a single, respective PLUN.

[109] Each VLUN is generally associated with a single, respective media unit.

[110] FIG. 7 illustrates the usage states of physical media units. Preferably, only those physical media units with a PLUN peripheral device type of "Direct Access" can be migrated to the virtual pool 134. Preferably, the physical media unit represents the entire address space of a physical LUN (PLUN). In one embodiment, all physical media units are initially created in the native pool 132 to ensure that their existing user data is preserved. Prior to migrating a physical media unit to the virtual pool 134, its usage state should preferably be "Unused" and its service state should preferably be "Online."

[111] Like all media units in the virtual pool 134, if a physical media unit is exported to a host by mapping it to a VLUN or it is used as a building block for other media units, its usage state is set to "Used." It will preferably remain in that state until the media unit is no longer mapped to a VLUN and it is not used as a building block for other media units.

[112] In one embodiment of the invention, if the physical LUN is deleted and the physical media unit is in the "Unused" state, the corresponding physical media unit will also be deleted. If the physical media unit is in the "Used" state when the PLUN is deleted, the physical media unit will not be deleted, but its service state will enter the "Offline" state. Should the PLUN be rediscovered on the same or a different port of the storage server 100, its corresponding physical media unit will return to the "Online" state.

[113] FIG. 7 shows three physical media units. The first physical media unit 130a has been exported to a host using a VLUN. The second physical media unit 130b has been placed into the "Reserved" state to prevent it from being automatically consumed by other media units. The third physical media unit 130c is in the "Unused" state and is available to be mapped to a host with a VLUN or to be used as a building block. Note that in the following sections, we will see physical media units used as building blocks to create other media units.

[114] The service states of the physical media unit follow the common behavior for media units in the virtual pool 134.

[115] FIG. 8 shows slice media units. A slice media unit represents a part of an address space of an existing media unit. When the user requests a media unit with certain characteristics of a size that is smaller than the media units with those characteristics, the media unit manager 120

allows an existing media unit to be sliced up. One slice media unit will be created to fulfill the request. The remainder of the address space of the media unit being sliced will be captured in a second slice media unit.

[116] If a Slice Media Unit is in the “Unused” state, it can be:

[117] 1. Carved up into smaller, adjacent slice media units; and

[118] 2. Merged with adjacent slice media units that are also in the “Unused” state.

[119] In one embodiment of the present invention, only a flat slicing model is supported.

Hierarchies of slices are not allowed. In other words, in such an embodiment it is not legal for a slice media unit to have another slice media unit as a child. This requirement can be relaxed in other embodiments.

[120] FIG. 8 shows a physical media unit 130 that has been carved up into three slice media units. The first slice media unit 130d has been exported to a host by mapping it to a VLUN. The second slice media unit 130e has been reserved for later use. The third slice media unit 130f is available to be used as a building block for other media units or to be exported to a host.

[121] According to one embodiment of the present invention, once the physical media unit 130 in FIG. 8 has been sliced, it enters the “Used” state. That means the physical media unit cannot be directly used as a building block for any other media units and it cannot be mapped to a VLUN. If all of the slice media units tied to the physical media unit are deleted, then the physical media unit will once again enter the “Unused” state.

[122] If in FIG. 8 the PLUN 0 went offline, that would result in the physical media unit 130 going offline, which, in turn, would result in the three slice media units 130d, 130e and 130f going offline.

[123] The service states of the slice media unit follow the common behavior for media units in the virtual pool 134. Note that moving media units from the virtual pool 134 to the native pool 132 are disallowed in one embodiment of the present invention.

[124] FIG. 9 shows a concatenation media unit 130g. A concatenation media unit represents an address space equal to the sum of the address spaces of its child media units. When the user requests a media unit with certain characteristics of a size that is larger than the media units with those characteristics, the media unit manager 120 allows existing, unused media units to be combined into a single, larger address space.

[125] According to one embodiment of the present invention, all media units that make up a concatenation media unit share the same performance characteristics. In addition, all characteristics are the same; e.g., non-raid devices are not concatenated with RAID devices, etc.

[126] A concatenation media unit should obey the following rules:

[127] 1. Minimum Number of Members is 2; and

[128] 2. Address space is the sum of address spaces of its members (each member size could be different).

[129] In FIG. 9, two physical media units 130 have been carved up into four slice media units 130h, 130i, 130j and 130k. One slice media unit 130h has been exported to a host using a VLUN. Another slice media unit 130k has been reserved for future use. The remaining two slice media units 130i and 130j have been used as building blocks to create the concatenation media unit 130g.

[130] Note that in the example of FIG. 9, if either PLUN 0 or PLUN 1 goes offline, its corresponding physical media unit will go offline as well, which in turn will make its corresponding slice media units 130h, 130i, 130j and 130k go offline, which will ultimately make the concatenation media unit 130g go offline.

[131] The service states of the concatenation media unit follow the common behavior for media units in the virtual pool 134.

[132] FIG. 10 shows a mirrored media unit 130m, also referred to as a RAID1 media unit. (RAID1 is a more specific term and is a subset of the term mirrored. In general, an embodiment of the present invention uses RAID1 media units, but the principles involved are equally applicable to embodiments using mirrored media units.) A RAID1 or mirrored media unit represents an address space with mirrored properties. When the storage configuration manager receives a request to create a reliable media unit, it uses the media unit manager 120 to create a mirrored media unit. Note that there are other ways to achieve a reliable media unit; for example, creating a physical media unit for a PLUN from a physical storage device that is an external mirrored or RAID1 array.

[133] According to one embodiment of the present invention, a mirrored media unit obeys the following rules:

[134] 1. Minimum number of members is 2;

[135] 2. Maximum number of members is set by a policy in the storage configuration manager;

[136] 3. Every member should be on a different physical disk;

[137] 4. The address space represented by the mirrored media unit is equal to that of the smallest member;

[138] 5. Every member should have their primary paths through different ports on a different line card in the storage server 100; and

[139] 6. All members should share the same performance characteristics, unless the user intentionally wants certain members to be faster to facilitate quick/preferred reads, or some of the members are remote. (In such a case, the mirrored media unit will be asynchronous, as described below with reference to FIG. 19.)

[140] In FIG. 10, the mirrored media unit 130m has three members; thus, a three-way mirror. Each of the members in FIG. 10 is a physical media unit 130 tied to a physical LUN.

[141] If a PLUN should go offline, its corresponding physical media unit 130 will go offline as well. The mirrored media unit 130m will detect this state change and enter a "Degraded" service state. At that point, it will attempt to swap in a spare and perform a reconstruction. Further information on mirrored media units and examples of the reconstruction process can be found in a subsequent portion of the present application.

[142] FIGS. 11-12 illustrate two techniques of using mirrored media units and slice media units to build "smaller" reliable media units. In FIG. 11, two physical media units 130 are combined into mirrored media unit 130n. This mirrored media unit 130n is then carved up using two slice media units 130o and 130p. One slice media unit 130o is mapped to a VLUN, and the other slice media unit 130p remains unused.

[143] In FIG. 12, two physical media units 130 are each partitioned into two slice media units 130q, 130r, 130s and 130t. Two mirrored media units 130u and 130v are then created using a slice media unit from each PLUN. Although this technique uses more media units than the previous technique, it can provide more flexibility and facilitate a priority algorithm during reconstruction.

[144] FIG. 13 shows a striped media unit 130w. As shown, the striped media unit has three members. The data associated with the striped media unit 130w is striped across the three member media units (i.e., the data is striped across the three corresponding PLUNs).

[145] FIGS. 14-19 provide more details on mirrored media units. A mirrored media unit is built from several media units that are referred to as members. The mirrored media unit

preferably maintains the “RAID-specific” states of its members. The media units that are members are not required to have knowledge that they are part of a mirror. Those members may simply know that they are “Used” and they are “Online.” So, for example, when a reconstruction is in progress according to one embodiment, only the mirrored media unit (and possibly a Reconstruct Process) need know which members are being reconstructed.

[146] Some members may be designated as preferred read members. Preferred read members may be determined based on the member’s access time or on load balancing criteria.

[147] Members may be synchronous or asynchronous. Synchronous members behave as write-through members. Asynchronous members behave as write-back members.

[148] In one embodiment, the present invention uses the following strategies concerning spare members. If a member is removed and no writes occur to the mirrored media unit before the member is returned, the member can be added back in. This uses a delayed spare swap until a write occurs or some timeout occurs. If a valid spare does not presently exist, then the storage server 100 may look at other spares in the spare pool and choose to intelligently create a concatenated spare out of it whose size would fit. If the valid spare size is bigger than the one needed, then the storage server 100 may slice the spares. Global spares may be made available to all mirrored media units. According to another embodiment, the storage server 100 may make global spares available depending upon each customer account. Finally, dedicated spares may be assigned to a specific mirrored media unit.

[149] According to one embodiment, the following long-running processes are associated with a mirrored media unit: verify, reconstruct, and snapshot/synchronize/write logging. The verify process verifies that the data across the members is in synch. The reconstruct process, after a spare is swapped in to replace a failed member, restores the data on such new member.

[150] The snapshot/synchronize/write logging process operates as follows. If a member is temporarily removed for a snapshot, all writes are logged. When the member is returned, it is quickly resynchronized. If the write log is full, the member is reconstructed.

[151] A mirrored media unit may maintain the “RAID-specific” states of its members. According to one embodiment, an associative class may be used to encapsulate information on the member relationship. According to another embodiment, a set of RAID properties may be appended to the member media units. In either embodiment, the media units that are members need have no knowledge that they are part of a mirrored media unit. In other words, a member

may simply know that it is “Used” and “Online.” It does not need to know that it is being reconstructed.

[152] When all of the members of a mirrored media unit are online, the mirrored media unit is in the Online-Fully Redundant state. If any of the members are not online, but the mirrored media unit is still providing redundancy using at least two online members, the mirrored media unit is in the Online-Degraded state. If the mirrored media unit is down to one online member, it enters the Online-Critical state.

[153] The following are examples of RAID-specific states of the members:

[154] 1. Member Online;

[155] 2. Member Offline;

[156] 3. Member Re-Synchronizing; and

[157] 4. Member Reconstructing.

[158] FIG. 14 shows an example of a process 200 of reconstructing a mirrored media unit according to one embodiment. In step 200a, the media unit manager 120 creates a mirrored media unit 204 using two physical (disk) media units 202a and 202b. The media unit manager 120 also sets up a global spare pool with a disk media unit 202c. The media unit manager 120 may use a subprocess/subcomponent/subprogram called a RAID manager to perform these functions.

[159] In step 200b, the media unit manager 120 exports the mirrored media unit 204 to a host using a VLUN.

[160] In step 200c, the PLUN associated with the disk media unit 202b fails. The mirrored media unit 204 is critical because it has a single point of failure.

[161] In step 200d, the mirrored media unit 204 has no dedicated spares, a spare (the disk media unit 202c) is retrieved from the global spare pool. The media unit manager 120 begins reconstruction of the data of the mirrored media unit 204 onto the disk media unit 202c. After reconstruction, the mirrored media unit returns to the online state.

[162] Although FIG. 14 gives a specific example for a mirrored media unit with two members, the process is also applicable to mirrored media units with more members.

[163] FIG. 15 shows an example of a process 210 of reconstructing a mirrored media unit that is made from slice media units, according to one embodiment. In step 210a, the media unit manager 120 creates two mirrored media units 216a and 216b across two disk media units 212a

and 212b that are sliced into four slice media units 214a, 214b, 214c and 214d. The mirrored media unit 216a is used, being mapped to a host using a VLUN. The mirrored media unit 216b is unused.

[164] In step 210b, the PLUN corresponding to the disk media unit 212b fails. This removes the availability of the slice media unit 214b from the mirrored media unit 216a, which enters a critical service state as it has only a single point of failure (i.e., the slice media unit 214a, corresponding to the disk media unit 212a, corresponding to the single remaining PLUN).

[165] In step 210c, a spare is retrieved and mapped to a new disk media unit 212c. The media unit manager 120 starts reconstruction on the disk media unit 212c.

[166] Although the mirrored media unit 216b is shown in FIG. 15, it is in an unused state. Thus, it does not need to be involved in the process 210. However, if the mirrored media unit 216b is mapped to a host using a VLUN or is otherwise in the used state, since it is virtualized, then the media unit manager 120 in step 210c can reconstruct the mirrored media unit 216b in parallel with the reconstruction of the mirrored media unit 216a. Alternatively, the media unit manager 120 may also assign a different priority to the reconstruction of the mirrored media unit 216b versus that of the mirrored media unit 216a.

[167] FIG. 16 shows an example of a process 220 of using a mirrored media unit to create a snapshot media unit, according to one embodiment. In step 220a, the media unit manager creates a mirrored media unit 224a using three members, physical media units 222a, 222b and 222c. (To create a snapshot, generally three or more members are preferred, as will be made apparent in subsequent steps.)

[168] In step 220b, the third member (physical media unit 222c) of the mirrored media unit 224a is temporarily disassociated therefrom and is associated with a snapshot media unit 224b. The mirrored media unit 224a is degraded and can still withstand a single point failure, as it still has two members. The snapshot media unit 224b may be mapped to a VLUN, exported to a host, etc.

[169] In step 220c, the physical media unit 222c is re-associated with the mirrored media unit 224a. The physical media unit 222c is then resynchronized. If the resynchronization buffer of the storage server 100 is full, a reconstruction may be performed.

[170] FIG. 17 shows an example of a process 230 of non-RAID to RAID conversion and adding RAID members. In step 230a, there is no RAID media unit; the media unit manager directly exports the physical media unit 232a to a VLUN.

[171] In step 230b, the media unit manager creates a mirrored media unit 234 using the physical media unit 232a as its source. The second member of the RAID, the physical media unit 232b, is reconstructed. The array is critical because it has a single point of failure.

[172] In step 230c, reconstruction is complete and the data is now mirrored. The mirrored media unit 234 is in the online service state (normal operation).

[173] In step 230d, a third member (physical media unit 232c) is added and reconstruction is started on it. The mirrored media unit 234 is in a degraded state until reconstruction is completed. At that point, the mirrored media unit 234 is in the online service state.

[174] FIG. 18 shows an example of a process 240 of VLUN migration. VLUN migration describes associating an existing VLUN with a new PLUN. In step 240a, the PLUN 242a is associated with the physical media unit 244a, which is exported to the VLUN 248. The physical media unit 244b is associated with the PLUN 242b.

[175] In step 240b, the VLUN 248 is associated with a mirrored media unit 246. The mirrored media unit 246 is associated with the physical media units 244a and 244b. The data is reconstructed on the physical media unit 242b. The mirrored media unit 246 has a service state of "migrate."

[176] In step 240c, the physical media unit 244b is exported to the existing VLUN 248. The mirrored media unit 246 is no longer needed and may be dissolved.

[177] A similar process can be used for VLUN replication. In summary, steps 240a and 240b proceed as described above. Once reconstruction is complete, the VLUN 248 is associated with the physical media unit 244a and a second VLUN (not shown) is created and associated with the physical media unit 244b.

[178] FIG. 19 shows examples of mirrored media units with synchronous and asynchronous members. A member may be updated asynchronously if its performance parameters differ from those of the other members, if it is located at a different location from the other members, if the network administrators so desire, or for other reasons.

[179] In the example 250, three physical media units 252a, 252b and 252c are respectively associated with three PLUNs. A mirrored media unit 254 having two members is built using the

two physical media units 252a and 252b. A mirrored media unit 256 is built using the mirrored media unit 254 and the physical media unit 252c. The mirrored media unit 256 is asynchronous because the mirrored media unit 254 is updated synchronously and the physical media unit 252c is updated asynchronously. The asynchronous mirrored media unit 256 is mapped to a VLUN.

[180] Although in the example 250 the asynchronous mirrored media unit 256 is in effect a “third level” in the media unit hierarchy, an asynchronous media unit can also be created at another level. For example, an asynchronous media unit can be built from two or more physical media units (the “first level” in the media unit hierarchy), in which case such asynchronous media unit would be in the “second level” in the media unit hierarchy.

[181] In the example 260, four physical media units 262a, 262b, 262c and 262d are respectively associated with four PLUNs. A first mirrored media unit 264a having two members is built using the two physical media units 262a and 262b, and a second mirrored media unit 264b having two members is built using the two physical media units 262c and 262d. Another mirrored media unit 266 is built using the two mirrored media units 264a and 264b. The mirrored media unit 266 is asynchronous because the mirrored media units 264a and 264b are updated asynchronously. The asynchronous mirrored media unit 266 is mapped to a VLUN.

[182] FIGS. 20-21 provide more details concerning snapshots.

[183] FIG. 20 shows an example of a process 270 of creating a snapshot by adding a full member. In step 270a, the media unit manager exports a slice media unit 272a to a VLUN 274a.

[184] In step 270b, the media unit manager creates another media unit 272b that is the same as the slice media unit 272a. The media unit manager creates a temporary mirrored snapshot media unit 278a with the two slice media units 272a and 272b as members. The media unit manager issues instructions for the temporary mirrored snapshot media unit 278a to reconstruct the data of the original slice media unit 272a onto the new slice media unit 272b. Note that in one embodiment of the present invention, any read request above the high water mark is sent to the original slice media unit 272a (until reconstruction has been completed).

[185] Then the media unit manager creates a media unit 276a of type WithSnap to export the original media unit 272a. Note that in one embodiment of the present invention, write requests are sent to the mirrored snapshot media unit 278a since the data above the high water mark should be copied to the reconstruction member and marked before it is executed.

[186] The temporary mirrored snapshot media unit 278a is exported to a VLUN 274b.

[187] In step 270c, after completion of the reconstruction, the media unit 272a is mapped to the VLUN 274a, and the media unit 272b is mapped to the VLUN 274b.

[188] In step 270d, prior to reconstruction of the first snapshot, the media unit manager creates a second mirrored snapshot media unit 278b. The mirrored snapshot media unit 278b has as its members the media unit 272a and a new media unit 272c. Refer to step 270b for the creation of the second mirrored snapshot media unit 278b. Note that in one embodiment of the present invention, the two reconstructions may be independent and that the high water mark of the two may differ as well.

[189] FIG. 21 shows an example of a process 280 of creating a snapshot using a reference-based algorithm. To simplify the figure, the PLUNs are not shown.

[190] In step 280a, the media unit manager exports the media unit 282a to a VLUN 284a. The media unit 282a may be of type slice, etc.

[191] In step 280b, the media unit manager creates a media unit 282b. The media unit 282b may be smaller than the media unit 282a. The media unit manager converts the served type of the media unit 282a to “snapped.” The media unit manager creates a snapshot media unit 286a and adds the media units 282a and 282b thereto. The time the snapshot media unit 286a is created is termed time t_0 . The media unit manager maps the snapshot media unit 286a to a VLUN 284b. The media unit manager instructs the snapshot media unit 286a to initialize a meta data structure on the media unit 282b.

[192] The media unit manager creates another media unit 288a of type “WithSnap” to export the original media unit 282a to the VLUN 284a.

[193] Note that in one embodiment of the present invention, the media unit manager sends write requests to the snapshot media unit 286a before they are executed since the data should be copied to the media unit 282b and marked in the meta data before it is modified. For example, consider the table 289a, in which blocks 3 and 6 have been modified after time t_0 .

[194] In step 280c, the media unit manager creates a second snapshot media unit 286b in accordance with step 280b above. The time the snapshot media unit 286b is taken is termed time t_0 , and the time reference of the original snapshot media unit 286a is now referred to as time t_1 .

[195] Consider the example shown in table 289b. Blocks 3 and 6 are modified after time t_0 as in table 289a. Blocks 1, 4 and 6 are modified after t_1 .

[196] FIG. 22 shows an example of a process 290 for merging media units. The media unit manager may merge media units to recover storage space, to defragment storage space, or when a VLUN is deleted, etc.

[197] In step 290a, a media unit 292a is sliced into three slice media units 294a, 294b and 294c. The slice media unit 294a is exported to a VLUN 296a, and the slice media unit 294b is exported to a VLUN 296b. The slice media unit 294c is unused.

[198] In step 290b, the VLUN 296b is deleted. The slice media unit 294b disappears, and the media unit 292a has only two slices 294a and 294c.

[199] In step 290c, the VLUN 296a is deleted. All the slice media units disappear, and the media unit 292a is unused.

[200] A noteworthy feature of an embodiment of the present invention is the hierarchical nature of media units. Essentially, media units 130 provide a mapping between PLUNs and VLUNs. For example, in FIG. 9 three hierarchies of media units map a VLUN to the PLUNs that store the data in the virtual storage space represented by the VLUN. The first level of the hierarchy is the physical media units which are directly mapped to the PLUNs.

[201] The second level of the hierarchy is the various types of media units that can be built from the first level media units. For example, in FIG. 9 the second-level media units are the four slice media units 130h, 130i, 130j and 130k. The second-level media units are indirectly mapped to the PLUNs by the first-level media units.

[202] The third level of the hierarchy is the various types of media units that can be built using at least one of the second-level media units. For example, in FIG. 9 the third-level media unit is the concatenation media unit 130g. As another example, in FIG. 19 the third-level media unit is the asynchronous mirrored media unit 256. The third-level media units are indirectly mapped to the PLUNs by the first-level and second-level media units.

[203] In conclusion, the embodiments of the present invention use virtual media units as building blocks to create higher-level types of storage, such as slices, concatenations, stripes and mirrors. The use of virtual media units allows building blocks to be created without user intervention or taking parts of the storage area network off-line. This greatly reduces the effort involved in managing a storage area network in the following ways.

[204] First, the embodiments of the present invention allow LUN migration, as shown in FIG. 18 and the associated description. In many existing systems, in order to move data from one

PLUN to another, a user may have to take the first PLUN offline, must manually copy the data to the second PLUN, then must bring the second PLUN online. The other users may be unable to read or write data to either PLUN until such process has been completed. The storage server 100 according to an embodiment of the present invention bypasses these inconveniences.

[205] Second, the embodiments of the present invention allow on-the-fly redundancy, as shown in FIG. 17 and the associated description. For example, if a host computer is currently provided 100 GB of storage at a certain reliability level, that reliability can be increased by adding another lower-level member to the higher-level media unit. The storage server 100 according to an embodiment of the present invention enables this to be performed conveniently and transparently to the various end users and host computers.

[206] Third, the embodiments of the present invention allow on-the-fly snapshots, as shown in FIG. 16 and the associated description. In many existing systems, the storage devices must be taken offline to make a snapshot, making them unavailable to the end users and host computers. The storage server 100 according to an embodiment of the present invention bypasses these inconveniences.

[207] Fourth, the embodiments of the present invention allow changing the attributes of media units. For media units in the virtual pool, the media unit manager can change them between RAID, non-RAID, sliced, striped and concatenated, etc. This may be done by simply building a new media unit with the desired attributes that uses the existing media units as building blocks.

[208] Fifth, the embodiments of the present invention allow for easy port migration. For example, if a storage device is to be moved and connected to another port, the host computers each do not have to be made aware of the move. Thus, the storage server 100 according to an embodiment of the present invention greatly simplifies the burden on the SAN administrators.

[209] Sixth, the embodiments of the present invention allow instant snapshots of data, as shown in FIG. 16 and the associated description. For example, for data stored in a mirrored media unit with three physical storage device members, an instant snapshot of the data can be obtained merely by breaking off one of the three legs of the mirrored media unit.

[210] Although many of the features of the present invention have been described as process steps, embodiments of the present invention implement these process steps using one or more of hardware, software, firmware and microcode in the storage server 100. Furthermore, the storage server 100 has embodiments that include one or more of subparts, subcomponents, subprocessors

and subroutines (e.g., the media unit manager 120) that implement these process steps or portions thereof.

[211] Although the above description has focused on specific embodiments, numerous variations and their equivalents are considered to be within the scope of the present invention, which is defined by the following claims. When the claims discuss process or method steps, the steps are intended to be performed in any order or simultaneously except where specifically noted.

WHAT IS CLAIMED IS:

1. A storage server connecting a plurality of host computers and a plurality of storage devices, said plurality of storage devices having associated therewith a plurality of physical logical unit numbers (PLUNs), said storage server including a processor that executes processing comprising:
mapping said plurality of PLUNs to a plurality of first media units; and
creating a second media unit from at least a portion of one or more of said plurality of first media units.
2. The storage server of claim 1, wherein said storage server uses said second media unit when transferring data between said plurality of host computers and said plurality of storage devices.
3. The storage server of claim 1, wherein said plurality of PLUNs are directly mapped to said plurality of first media units.
4. The storage server of claim 1, wherein said plurality of PLUNs are indirectly mapped to said plurality of first media units.
5. The storage server of claim 1, wherein said second media unit is at least one of a slice media unit, a concatenation media unit, a striped media unit, and a mirrored media unit.
6. The storage server of claim 1, wherein said plurality of first media units comprises a plurality of physical media units.
7. The storage server of claim 1, wherein said plurality of first media units includes at least one of a physical media unit, a slice media unit, a concatenation media unit, a striped media unit, and a mirrored media unit.
8. The storage server of claim 1, wherein said second media unit is an asynchronous mirrored media unit.

9. The storage server of claim 1, wherein said processor further executes processing comprising:
creating a third media unit from at least a portion of another one or more of said plurality of first media units; and
creating a fourth media unit from said second media unit and said third media unit.
10. The storage server of claim 9, wherein said second media unit is a first slice media unit, said third media unit is a second slice media unit, and said fourth media unit is a concatenation media unit.
11. The storage server of claim 9, wherein said second media unit is a first mirrored media unit, said third media unit is a second mirrored media unit, and said fourth media unit is a slice media unit.
12. The storage server of claim 9, wherein said second media unit is a first slice media unit, said third media unit is a second slice media unit, and said fourth media unit is a mirrored media unit.
13. The storage server of claim 9, wherein said second media unit is a first synchronous mirrored media unit, said third media unit is a second synchronous mirrored media unit, and said fourth media unit is an asynchronous mirrored media unit.
14. The storage server of claim 1, wherein said portion of said one or more of said plurality of first media units is a second portion of one of said plurality of first media units, and said second media unit is a slice media unit.
15. The storage server of claim 1, wherein said portion of said one or more of said plurality of first media units is all of two or more of said plurality of first media units, and said second media unit is a mirrored media unit.
16. The storage server of claim 1, wherein said plurality of first media units is two or more first media units, wherein said second media unit is a mirrored media unit, and wherein said processor further executes processing comprising:

disassociating one of said two or more first media units from said mirrored media unit; and

creating a third media unit from the disassociated first media unit, wherein said third media unit comprises a snapshot of said mirrored media unit.

17. The storage server of claim 16, wherein said processor further executes processing comprising:

re-associating the disassociated first media unit with said mirrored media unit, after said third media unit is created.

18. The storage server of claim 1, wherein said processor further executes processing comprising:

adding a third media unit associated with another of said plurality of PLUNs; associating said third media unit with said second media unit, wherein said second media unit is a mirrored media unit; and

reconstructing data of said second media unit on said another of said plurality of PLUNs via said third media unit.

19. The storage server of claim 18, wherein said third media unit is associated with said second media unit to provide a greater level of redundancy.

20. The storage server of claim 18, wherein said third media unit is associated with said second media unit to replace a failed media unit.

21. The storage server of claim 1, wherein said portion of one or more of said plurality of first media units comprises a first media unit associated with a first PLUN of said plurality of PLUNs, and wherein said processor further executes processing comprising:

adding a third media unit associated with at least a portion of a second PLUN of said plurality of PLUNs;

associating said third media unit with said second media unit, wherein said second media unit is a mirrored media unit;

reconstructing data of said second media unit on said second PLUN via said third media unit; and

disassociating said third media unit from said second media unit.

22. The storage server of claim 1, wherein said second media unit is a synchronous mirrored media unit, and wherein said processor further executes processing comprising:

creating a third media unit from said second media unit and another of said plurality of first media units, wherein said third media unit is an asynchronous mirrored media unit.

23. The storage server of claim 1, further comprising:

a plurality of processors, wherein said plurality of processors includes said processor, and wherein said processing is distributed among said plurality of processors.

24. The storage server of claim 1, wherein said processor further executes processing comprising:

exporting a first designated media unit of said plurality of first media units to a first virtual LUN;

reconstructing a second designated media unit of said plurality of first media units using said second media unit, wherein said second media unit is a mirrored snapshot media unit;

exporting said second media unit to a second virtual LUN; and

associating a third media unit with said first designated media unit, said first virtual LUN and said second media unit.

25. The storage server of claim 24, wherein said processor further executes processing comprising:

creating a fourth media unit from at least a portion of one or more of said plurality of first media units, wherein said fourth media unit is a second mirrored snapshot media unit;

associating said third media unit with said first designated media unit, said first virtual LUN, said second media unit and said fourth media unit;

reconstructing a third designated media unit of said plurality of first media units using said fourth media unit; and

exporting said fourth media unit to a third virtual LUN.

26. The storage server of claim 24, wherein after completion of reconstructing said second designated media unit, said processor further executes processing comprising:
exporting said first designated media unit directly to said first virtual LUN; and
exporting said second designated media unit directly to said second virtual LUN.

27. The storage server of claim 1, wherein said processor further executes processing comprising:
exporting a first designated media unit of said plurality of first media units to a first virtual LUN;
creating a second designated media unit of said plurality of first media units, wherein said second designated media unit is smaller than said first designated media unit;
associating said second media unit with said first designated media unit and said second designated media unit, wherein said second media unit is a snapshot media unit;
exporting said second media unit to a second virtual LUN;
creating a third media unit and exporting said third media unit to said first virtual LUN, wherein said third media unit is associated with said first designated media unit and said second media unit; and
creating a fourth media unit and exporting said fourth media unit to a third virtual LUN, wherein said fourth media unit is a second snapshot media unit, and wherein said fourth media unit is associated with said second media unit and a third designated media unit of said plurality of first media units.

28. The storage server of claim 1, wherein said second media unit is one of a plurality of slice media units associated with one of said plurality of first media units, wherein said processor further executes processing comprising:
deleting a first virtual LUN associated with a third media unit of said plurality of slice media units, and in response thereto deleting said third media unit; and
deleting a second virtual LUN associated with said second media unit, and in response thereto deleting said second media unit.

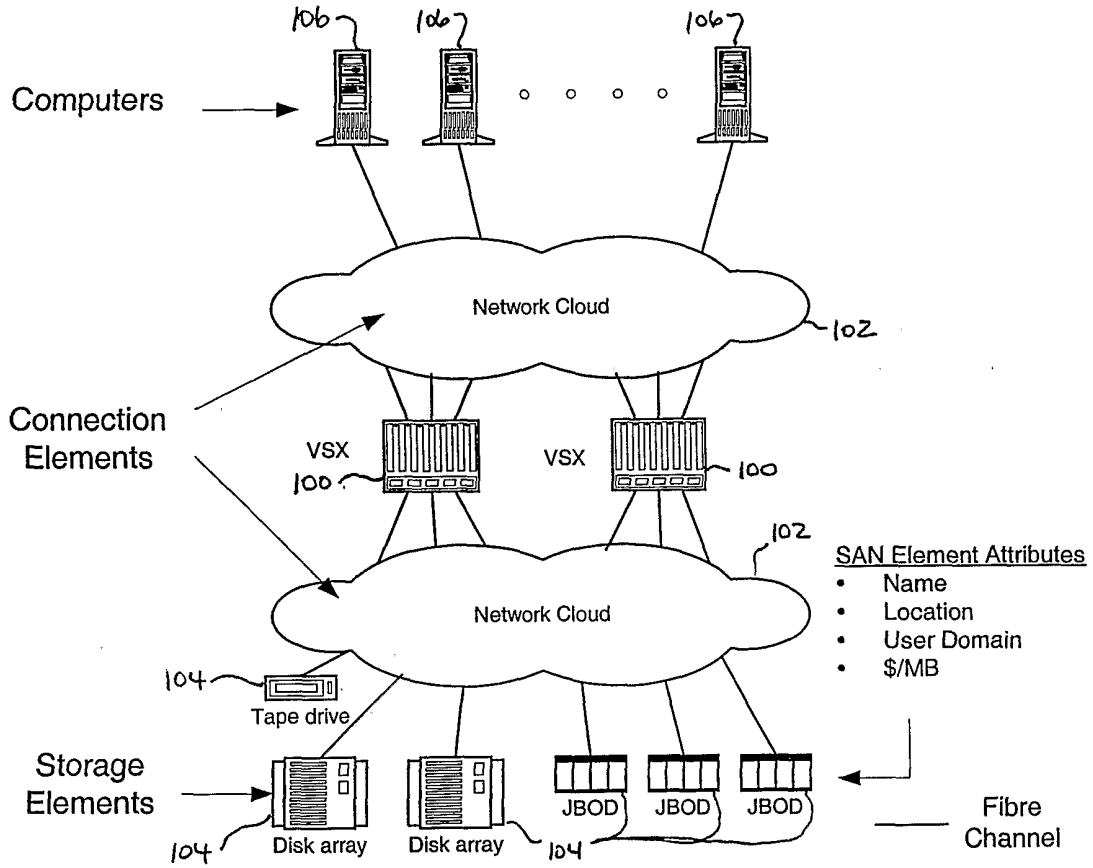


FIG. 1

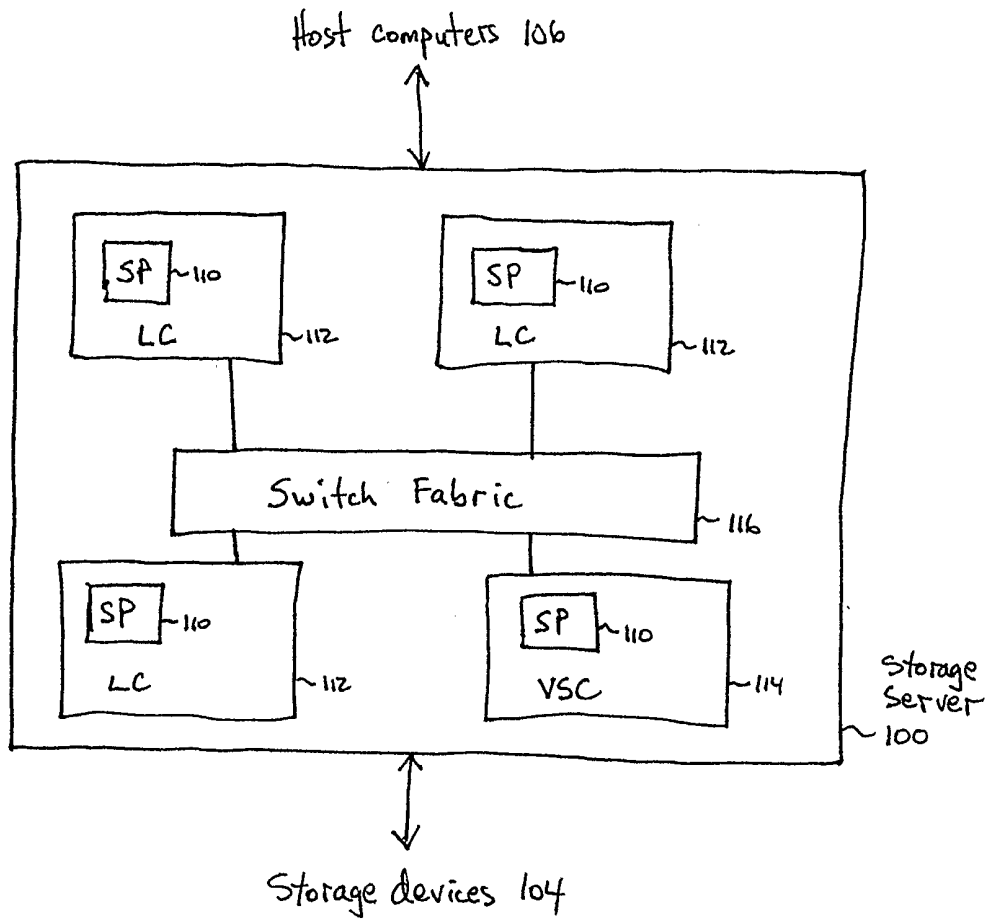


FIG. 2A

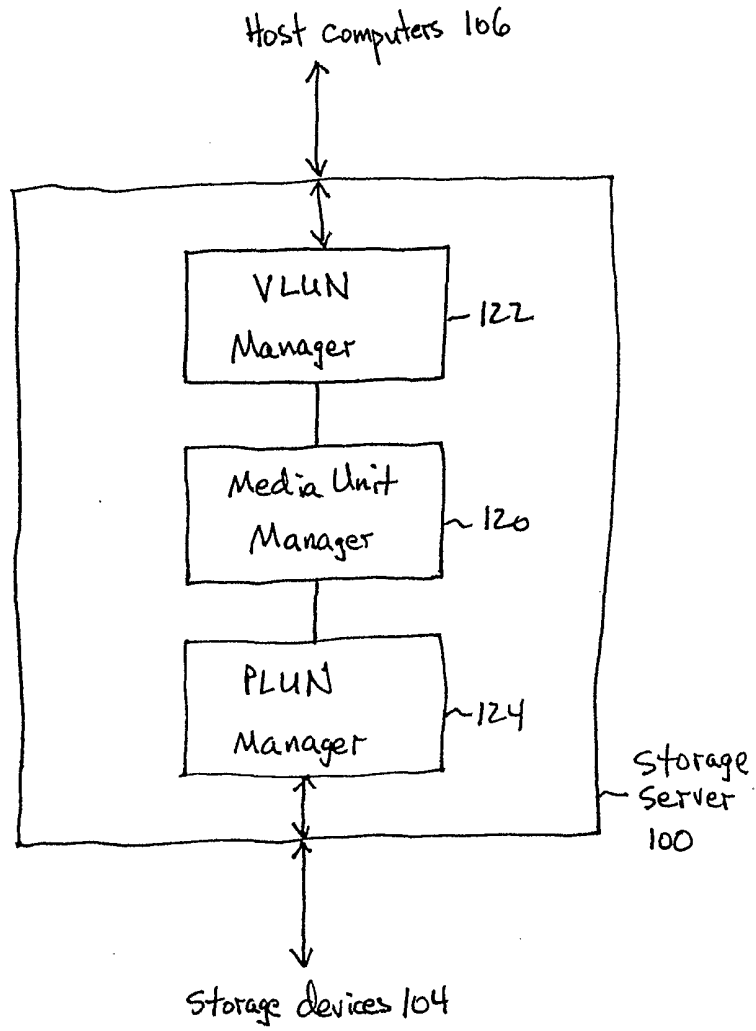


FIG. 2B

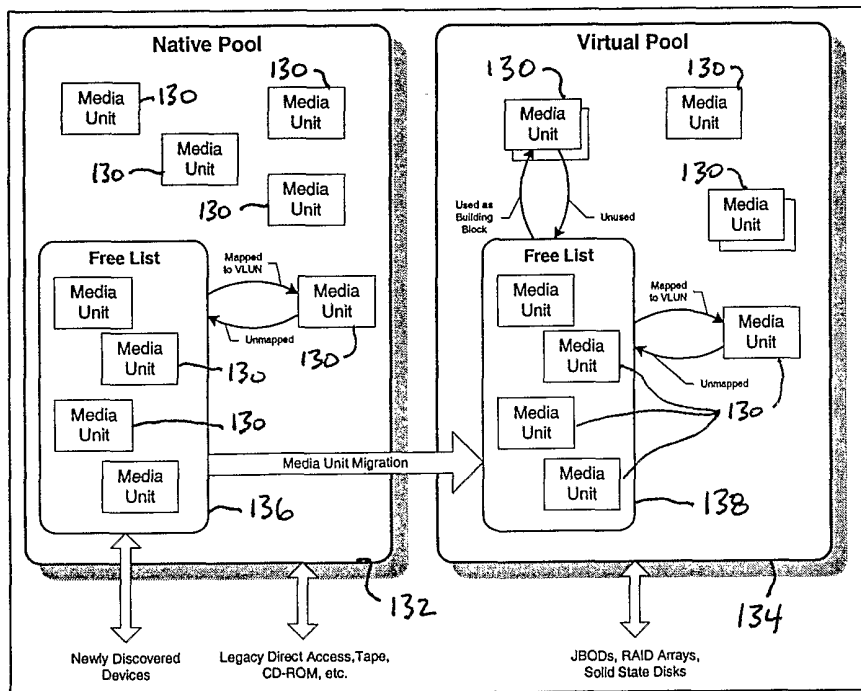


FIG. 3A

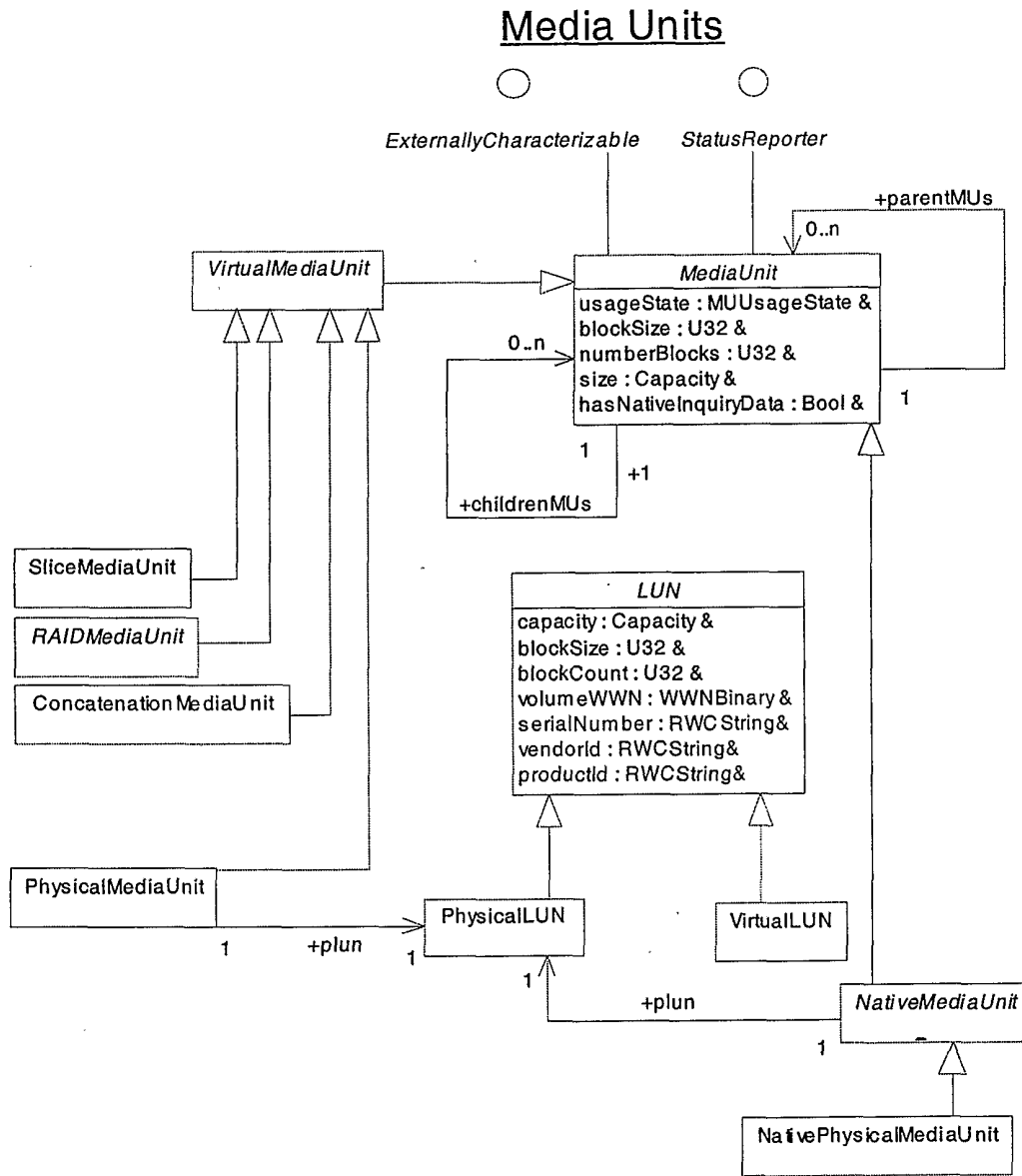


FIG. 3B

MU Usage States

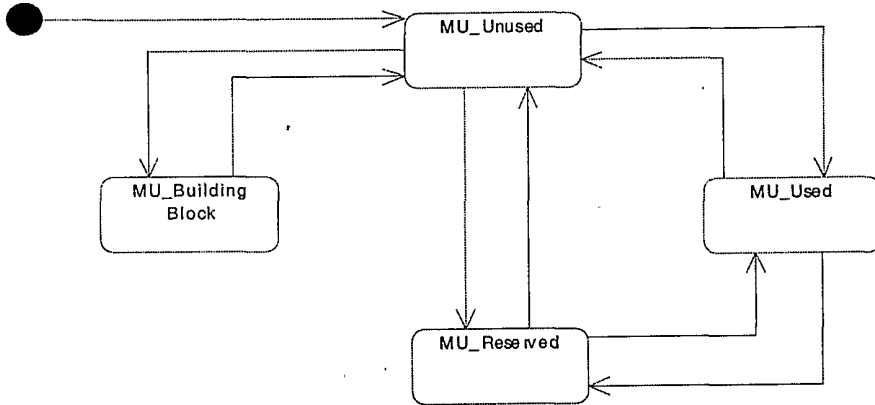


FIG. 4A

MU Service States

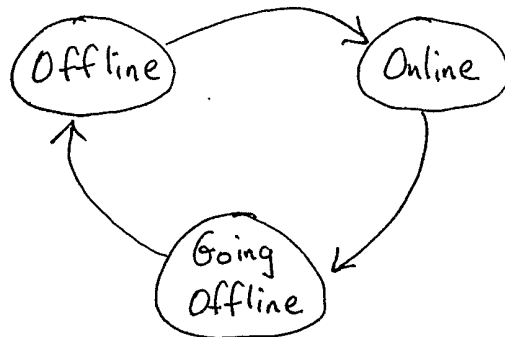


FIG. 4B

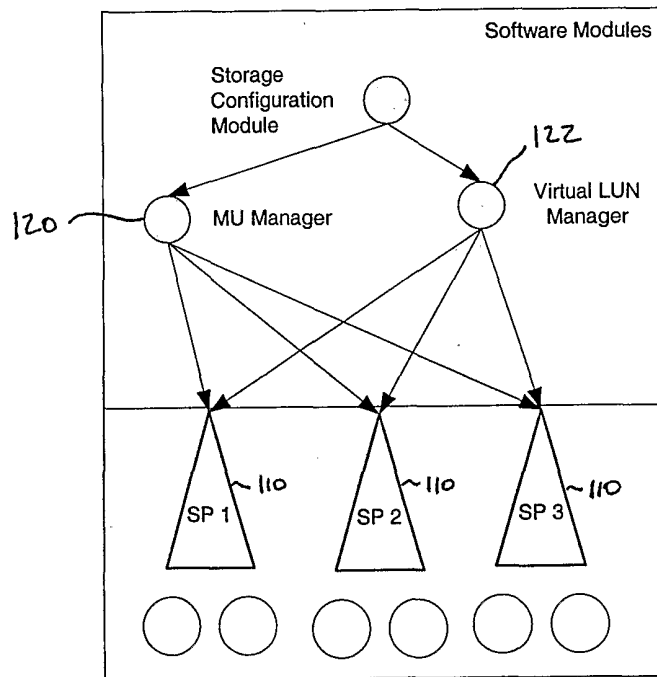


FIG. 5A

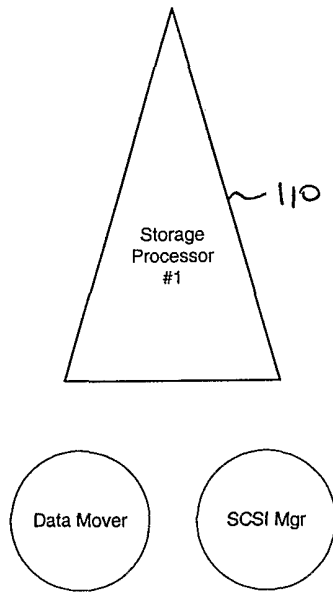


FIG. 5B

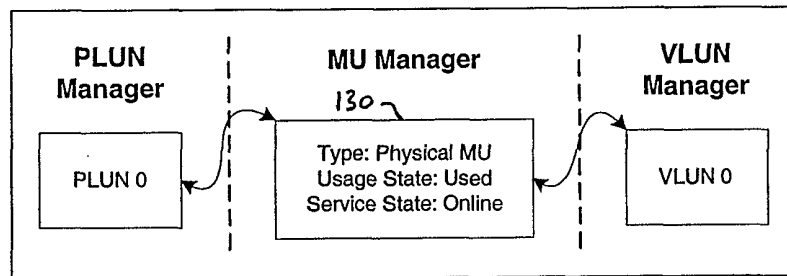


FIG. 6

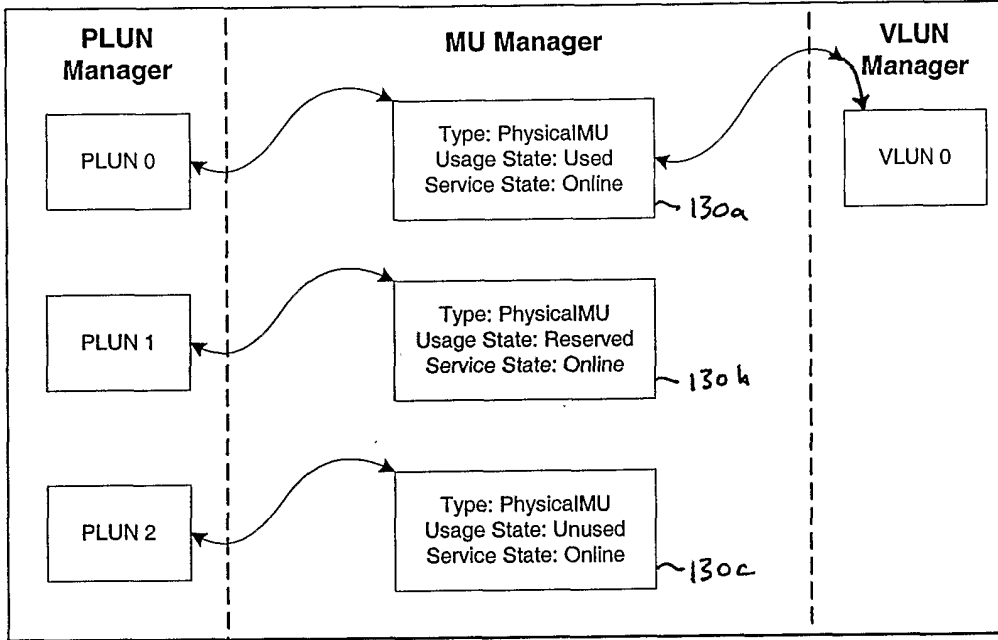


FIG. 7

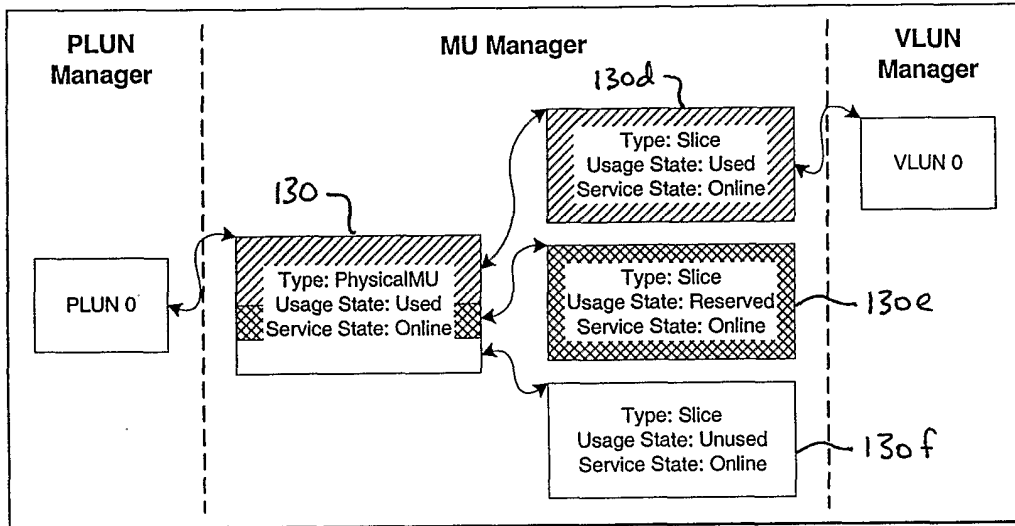


FIG. 8

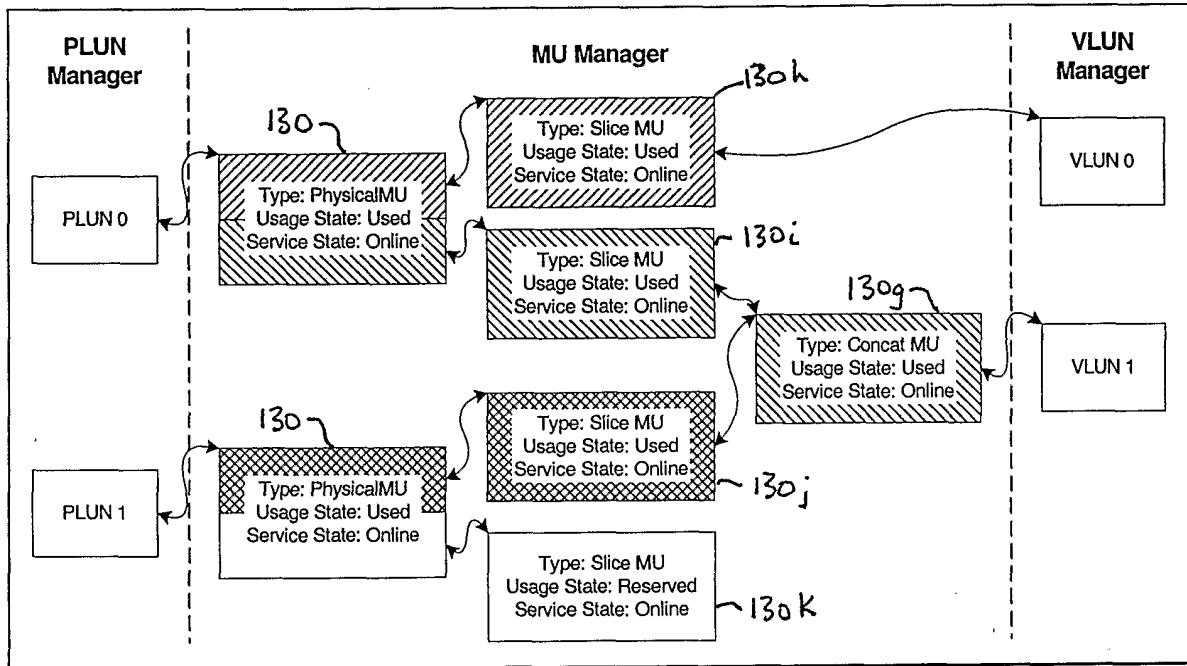
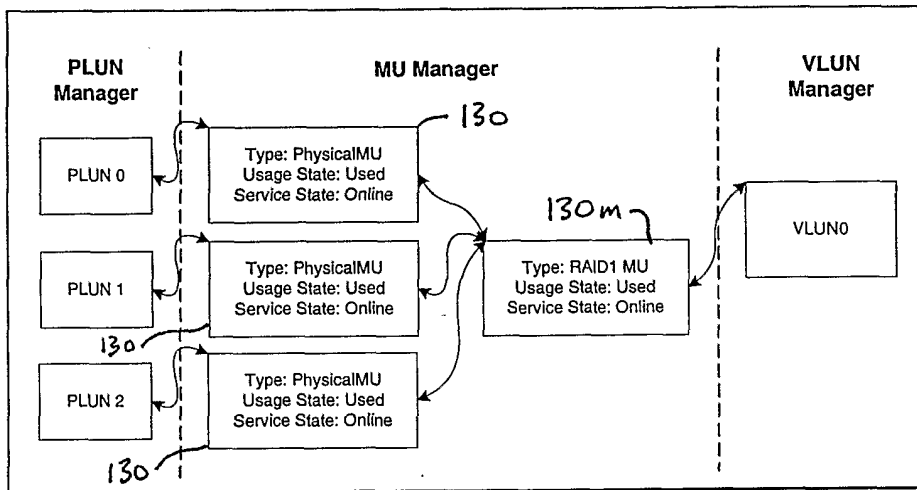


FIG. 9



F16.10

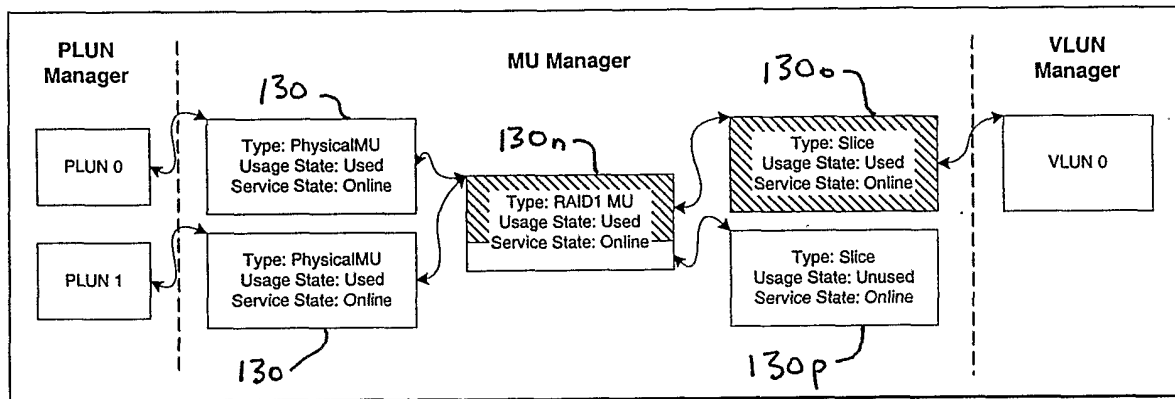


FIG. 11

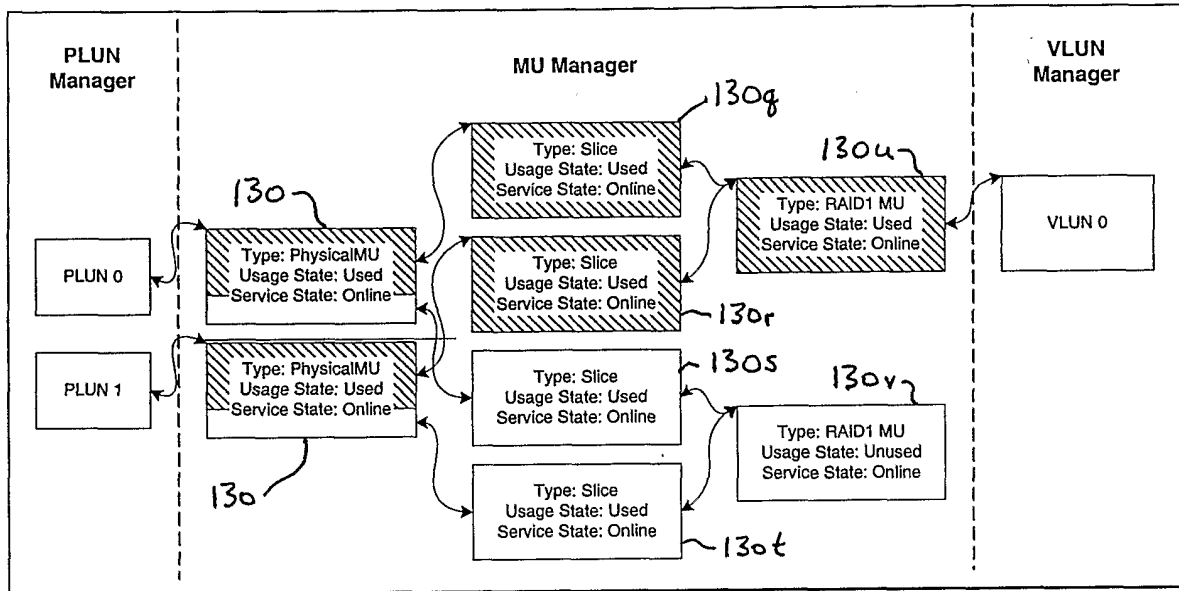


FIG. 12

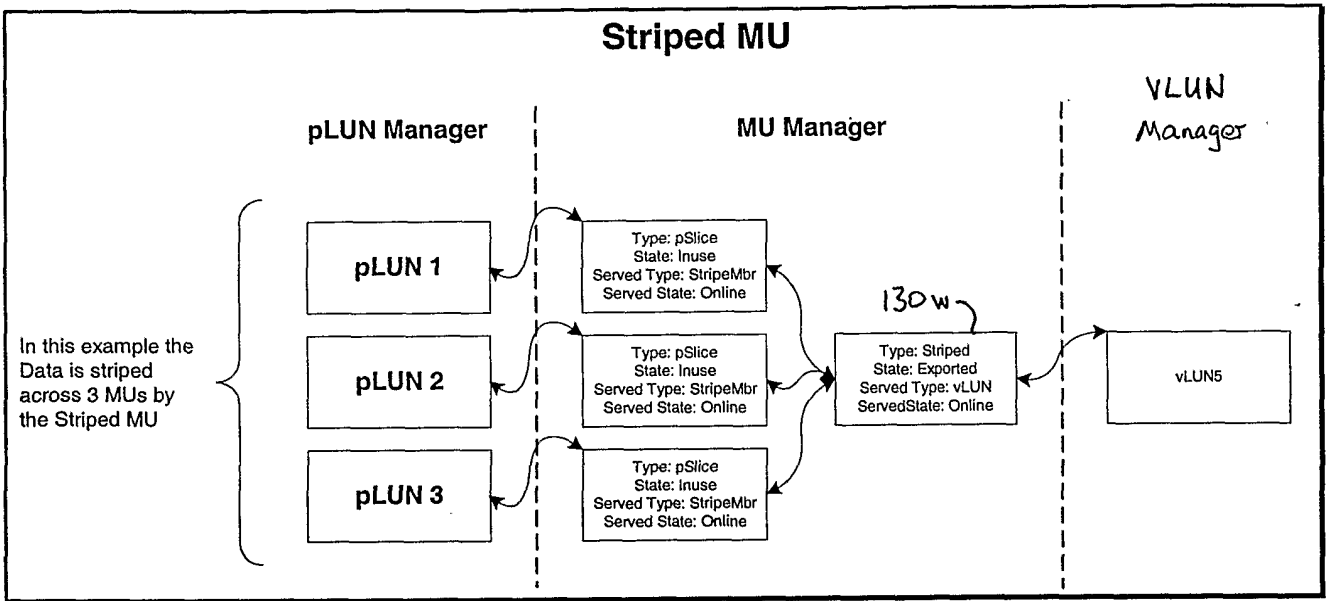
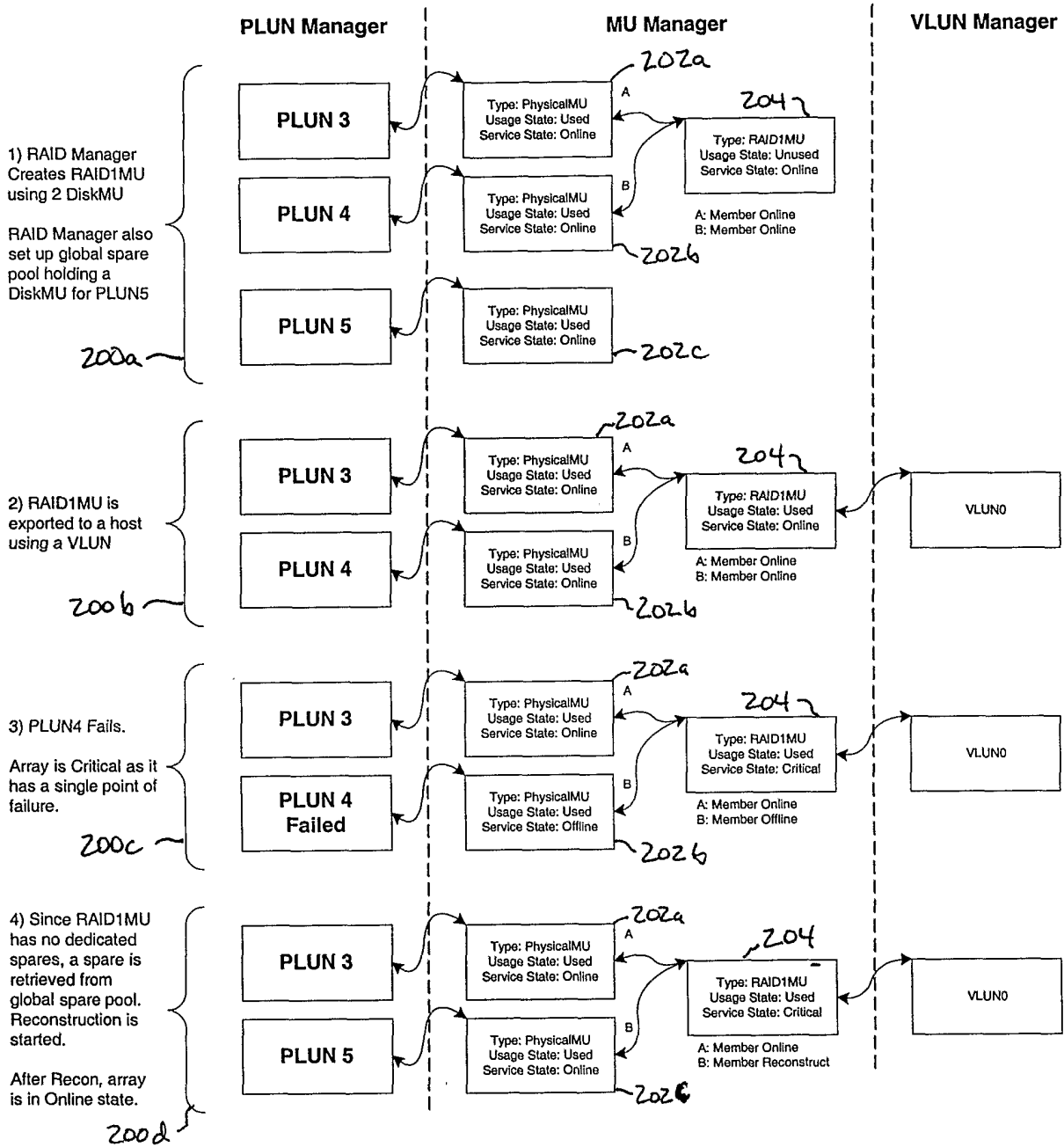


FIG. 13



200 ↗

FIG. 14

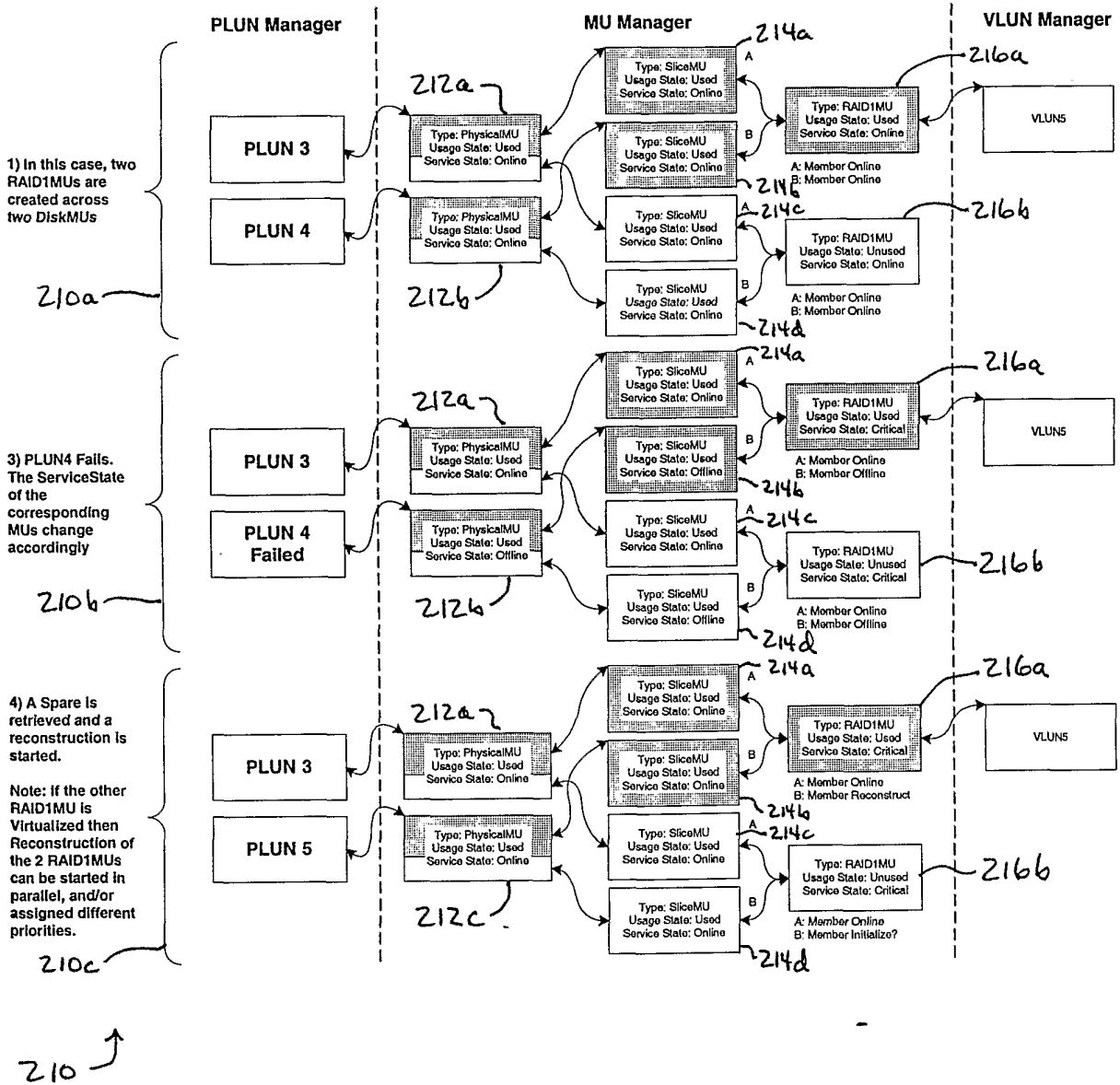


FIG. 15

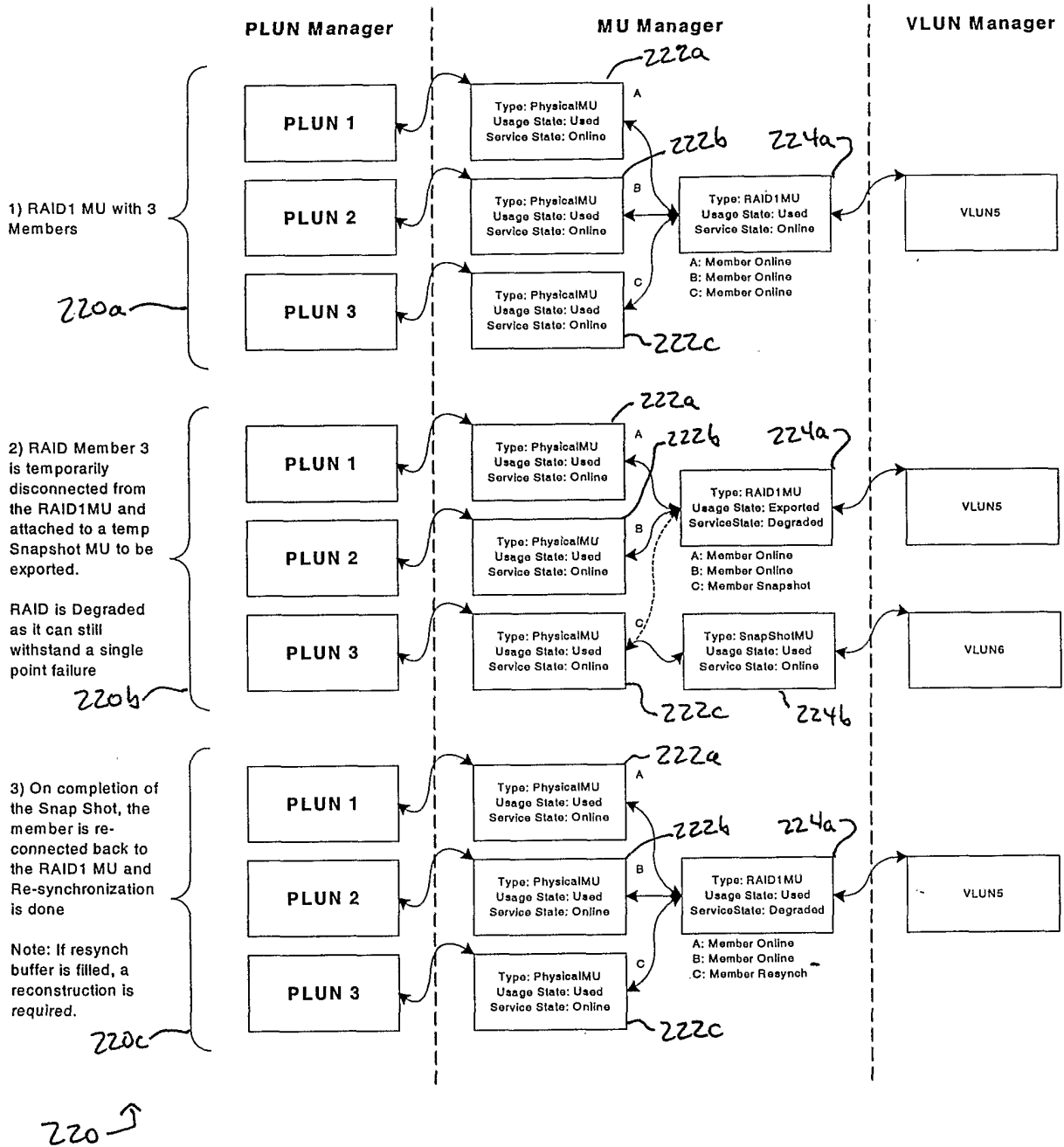
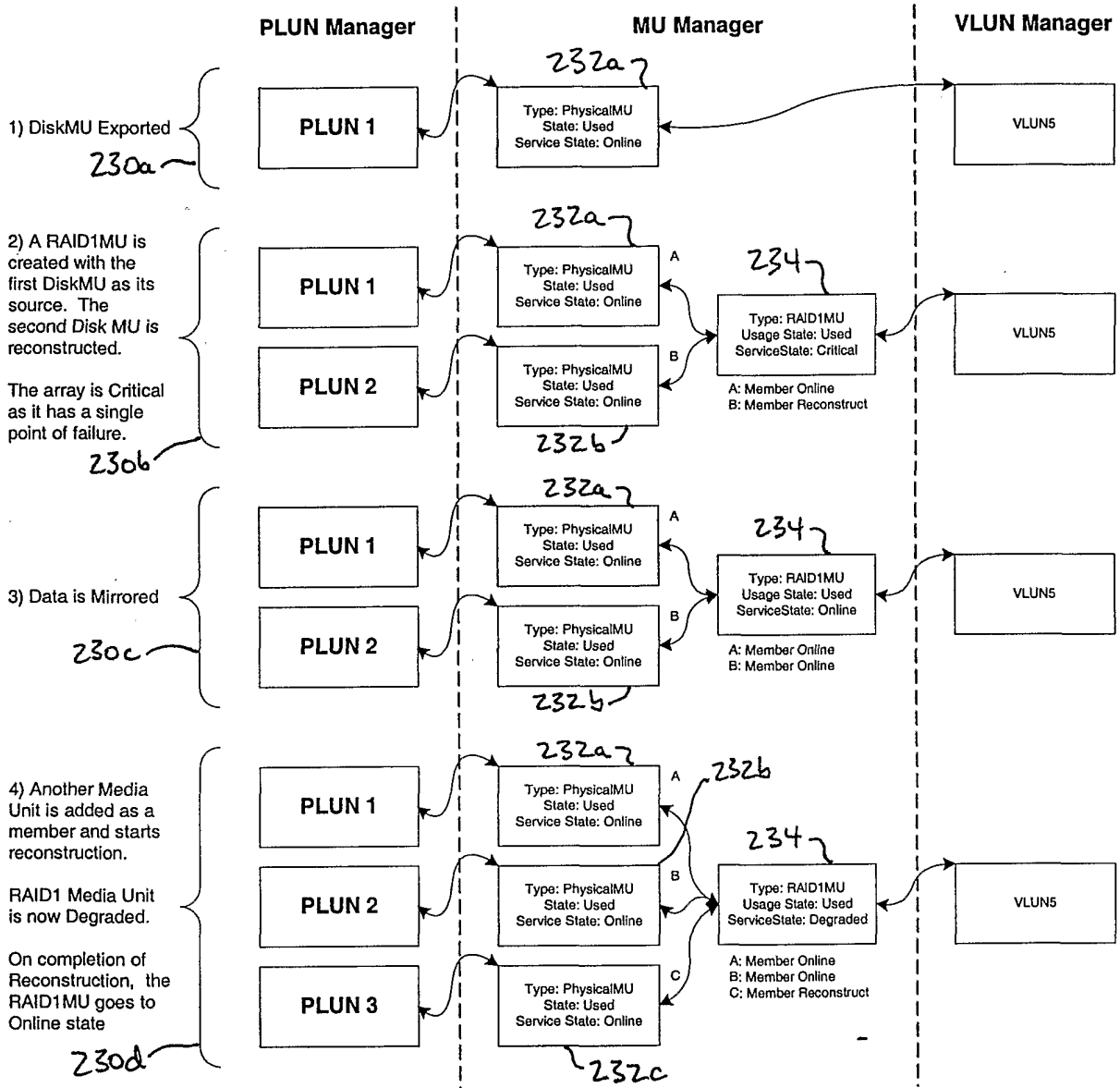


FIG. 16



230 →

FIG. 17

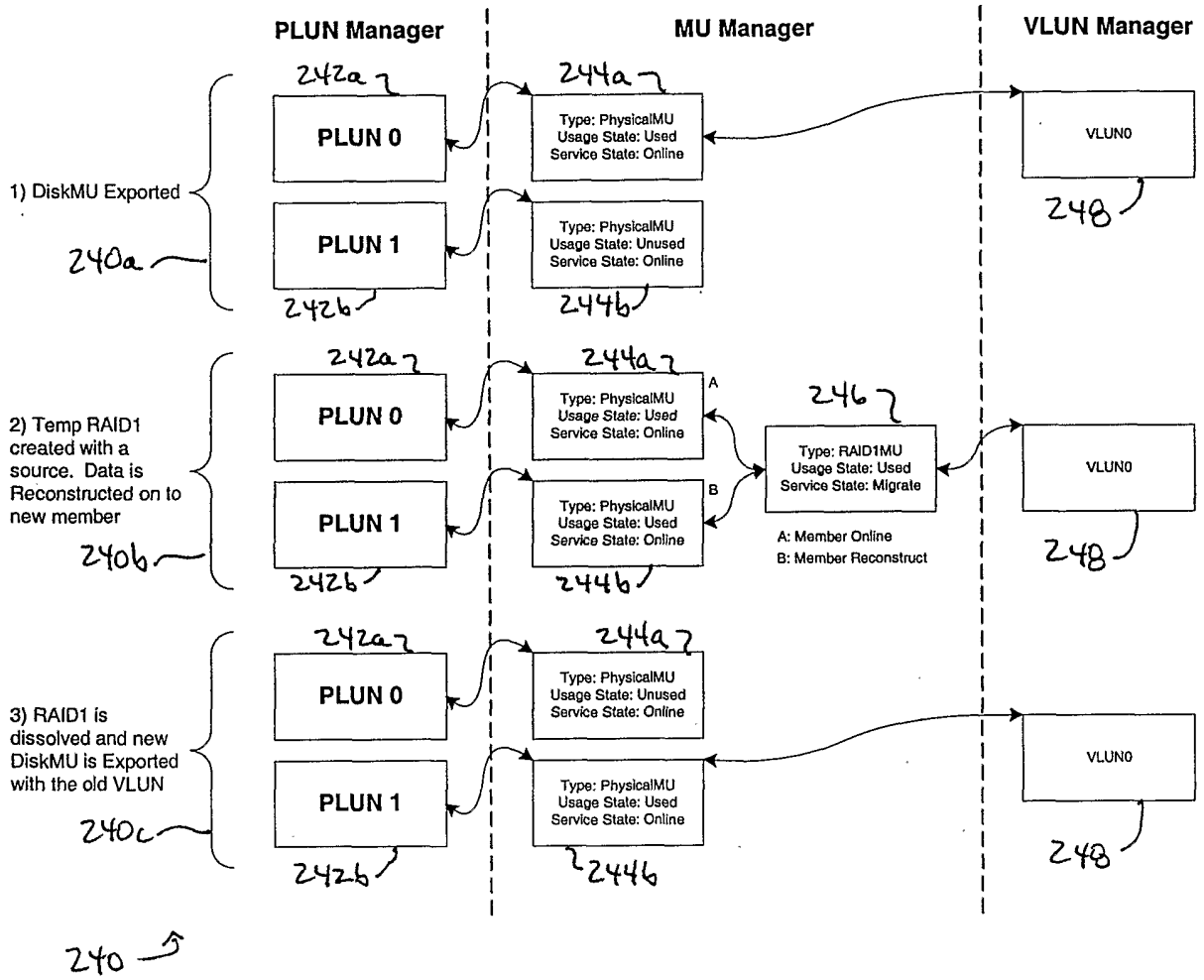


FIG. 18

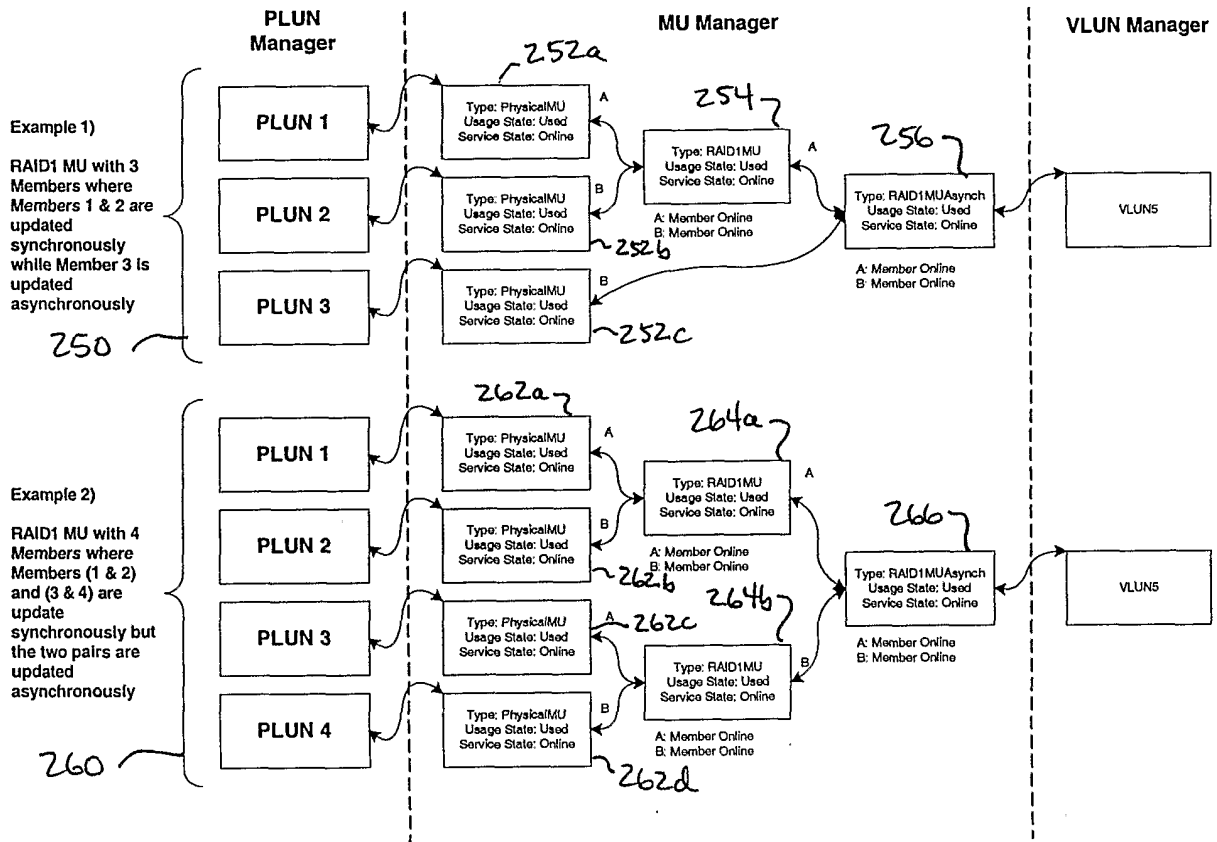


FIG. 19

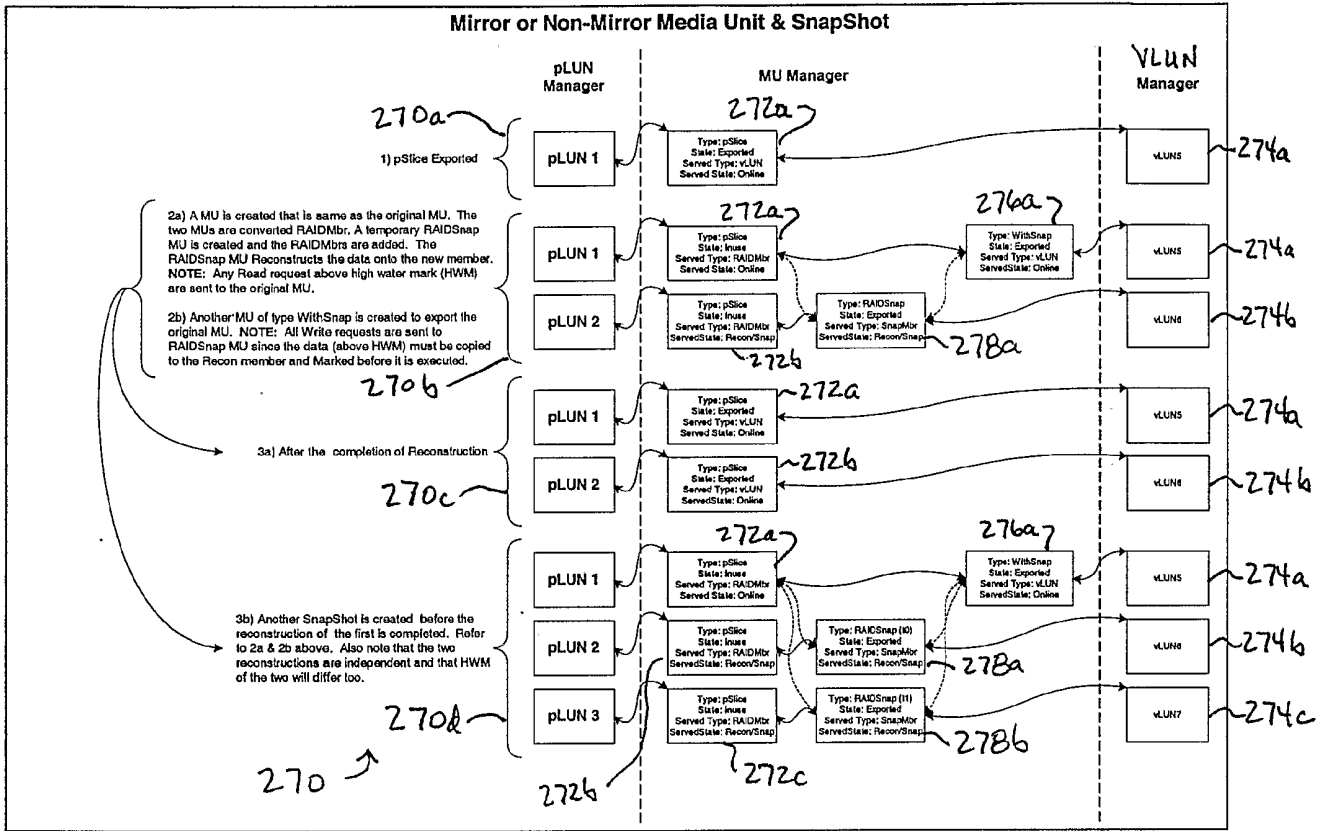
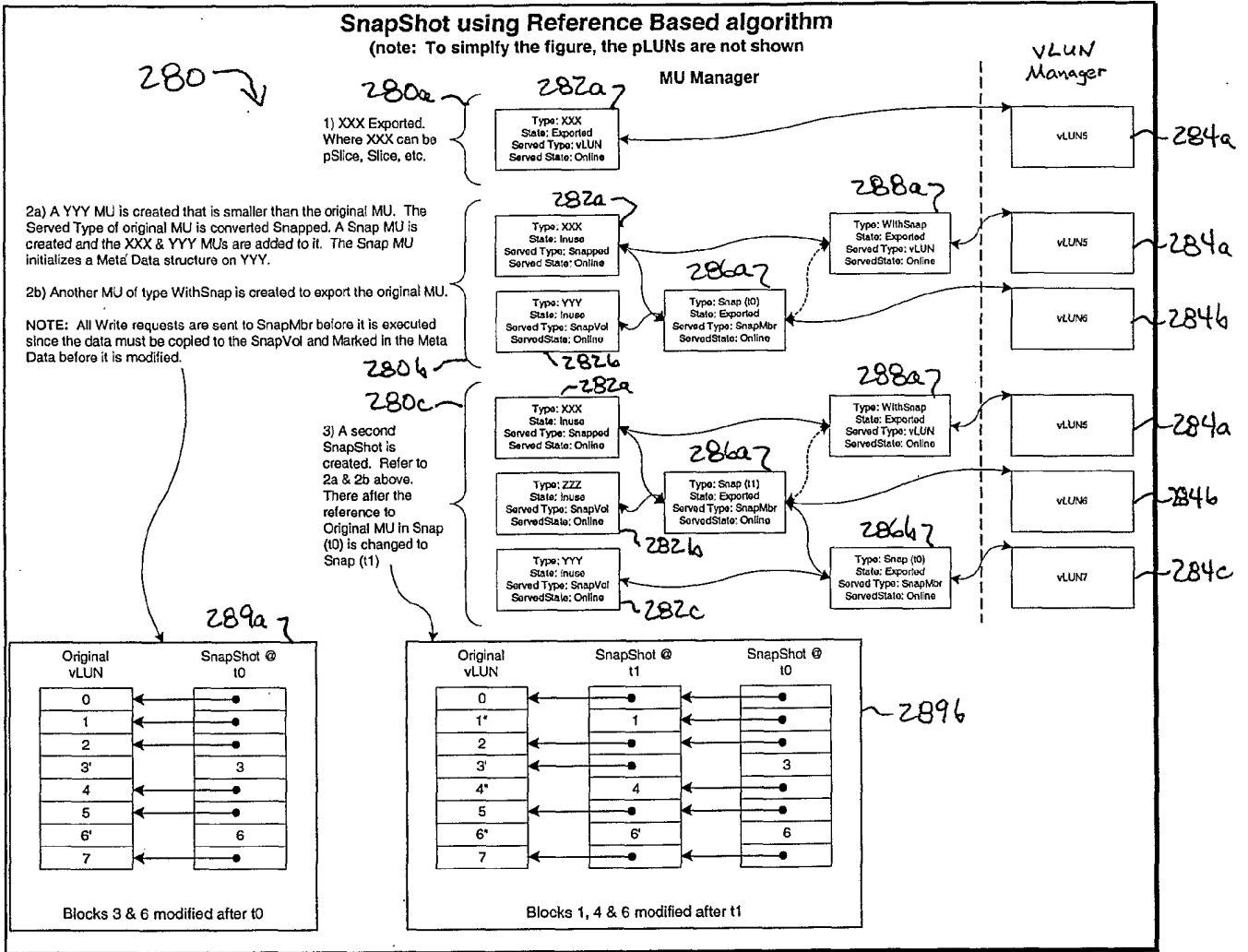


FIG. 20



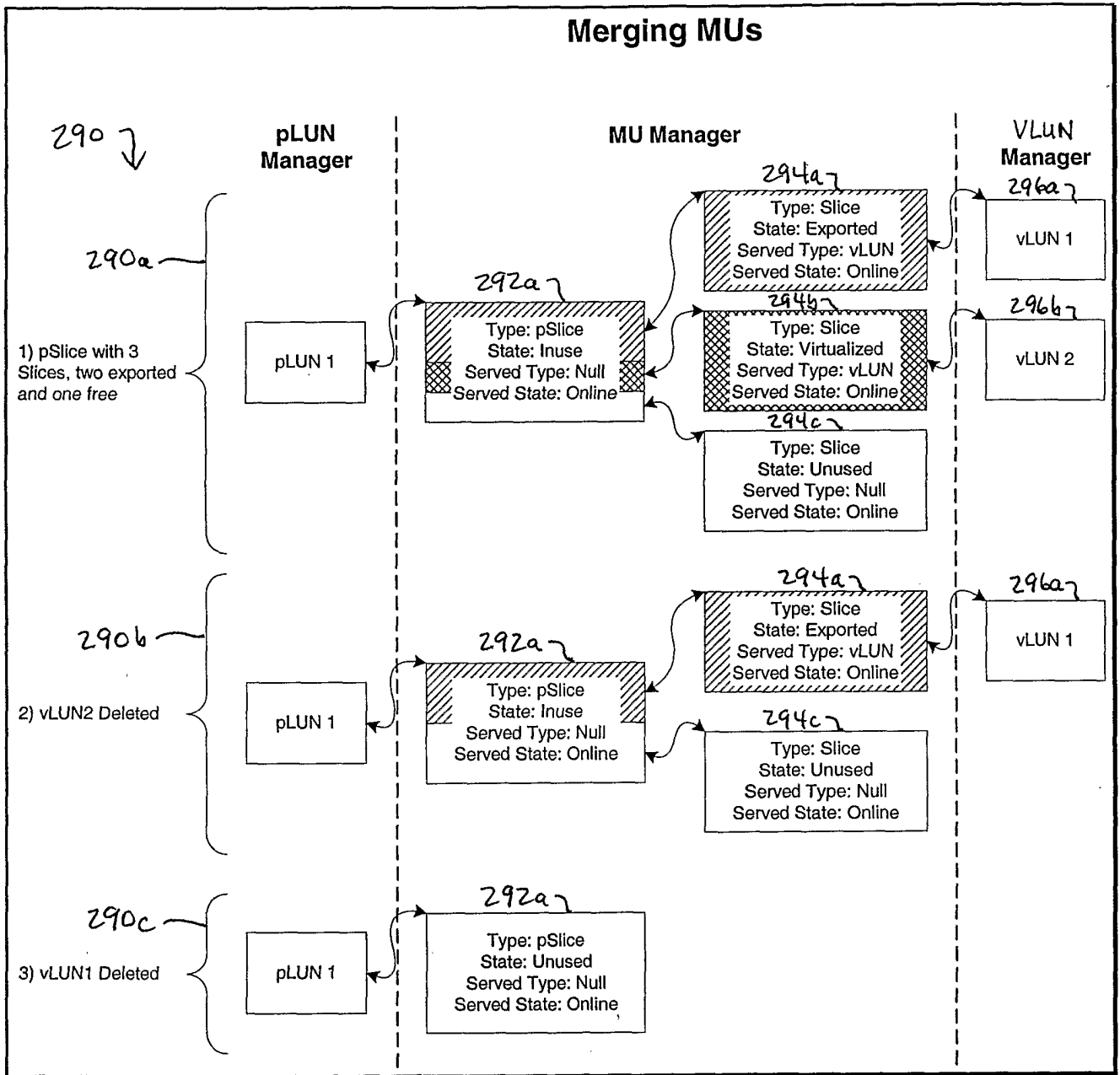


FIG. 22