(54) **Title**: RECOVERY OF APPLICATION FROM SNAPSHOT
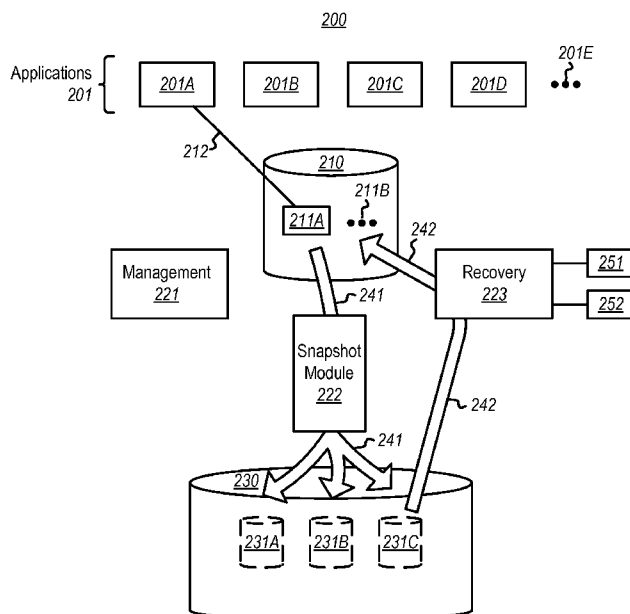
FIG. 2

(57) **Abstract**: The targeted recovery of application-spe-
cific data corresponding to an application without per-
forming recovery of the entire volume. The recovery is
initiated by beginning to copy the prior state of the con-
tent of an application-specific data container from a prior
snapshot to the application-specific data container in an
operation volume accessible by the application. However,
while the content of the application-specific data contain-
er is still being copied from the snapshot to the applica-
tion-specific data container, the application is still permit-
ted to perform read and write operations on the applica-
tion-specific data container. Thus, the application-specific
data container appears to the application to be fully ac-
cessible even though recovery of the content of the ap-
plication-specific data container is still continuing in the
background.

— as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))

Published:

— with international search report (Art. 21(3))

— before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))

RECOVERY OF APPLICATION FROM SNAPSHOT

BACKGROUND

[0001] Computers and computing systems have affected nearly every aspect of modern living. Computers are generally involved in work, recreation, healthcare, transportation, entertainment, household management, etc. Computer functionality is typically the result of computing systems executing software code. The computer uses a volume to store the software code, and also to store data.

[0002] Occasionally, the volume may become corrupted, or the volume itself may be damaged. In either case, the volume is rendered unusable. In order to prepare for such a failure event, a computing system may cause periodic snapshots of the volume to be taken. If a failure event occurs, the volume may be replaced if damaged, and a recovery module may facilitate recovery of the volume by copying back from the latest valid snapshot into the volume. Thus, despite failure of the volume, the volume is returned to a prior state, allowing computer functionality to return.

SUMMARY

[0003] In accordance with at least one embodiment described herein, the targeted recovery of application-specific data corresponding to an application is described. As an example, the application might be a virtual machine, while the application-specific data might be a virtual hard drive that corresponds to the virtual machine. The recovery of the application-specific data is performed without performing recovery of the entire volume.

[0004] The recovery is initiated by beginning to copy the prior state of the content of an application-specific data container from a prior snapshot to the application-specific data container in an operation volume accessible by the application. However, while the content of the application-specific data container is still being copied from the snapshot to the application-specific data container, the application is still permitted to perform read and write operations on the application-specific data container. Thus, the application-specific data container appears to the application to be fully accessible even though recovery of the content of the application-specific data container is still continuing in the background.

[0005] This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used as an aid in determining the scope of the claimed subject matter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006]   In order to describe the manner in which the above-recited and other advantages and features can be obtained, a more particular description of various embodiments will be rendered by reference to the appended drawings. Understanding that these drawings depict only sample embodiments and are not therefore to be considered to be limiting of the scope of the invention, the embodiments will be described and explained with additional specificity and detail through the use of the accompanying drawings in which:

[0007]   Figure 1 abstractly illustrates a computing system in which some embodiments described herein may be employed;

[0008]   Figure 2 schematically illustrates an example system in which targeted recovery of an application may occur without recovering the entire system in accordance with the principles described herein;

[0009]   Figure 3 illustrates a flowchart of a method for recovering application specific data corresponding to an application;

[0010]   Figure 4 illustrates a flowchart of a method for performing a read operation while a recovery operation is still ongoing; and

[0011]   Figure 5 illustrates a flowchart of a method for performing a write operation while a recovery operation is still ongoing.

DETAILED DESCRIPTION

[0012]   In accordance with embodiments described herein, the targeted recovery of application-specific data corresponding to an application without performing recovery of the entire volume is described. The recovery is initiated by beginning to copy the prior state of the content of an application-specific data container from a prior snapshot to the application-specific data container in an operation volume accessible by the application. However, while the content of the application-specific data container is still being copied from the snapshot to the application-specific data container, the application is still permitted to perform read and write operations on the application-specific data container using the principles described further below. Thus, the application-specific data container appears to the application to be fully accessible even though recovery of the content of the application-specific data container is still continuing in the background.

[0013]   Some introductory discussion of a computing system will be described with respect to Figure 1. Then, the targeted recovery will be described in the order provided above with respect to Figures 2 through 5.

2

[0014]   Computing systems are now increasingly taking a wide variety of forms. Computing systems may, for example, be handheld devices, appliances, laptop computers, desktop computers, mainframes, distributed computing systems, or even devices that have not conventionally been considered a computing system. In this description and in the claims, the term "computing system" is defined broadly as including any device or system (or combination thereof) that includes at least one physical and tangible processor, and a physical and tangible memory capable of having thereon computer-executable instructions that may be executed by the processor. The memory may take any form and may depend on the nature and form of the computing system. A computing system may be distributed over a network environment and may include multiple constituent computing systems.

[0015]   As illustrated in Figure 1, in its most basic configuration, a computing system 100 includes at least one processing unit 102 and computer-readable media 104. The computer-readable media 104 may conceptually be thought of as including physical system memory, which may be volatile, non-volatile, or some combination of the two. The computer-readable media 104 also conceptually includes non-volatile mass storage. If the computing system is distributed, the processing, memory and/or storage capability may be distributed as well.

[0016]   As used herein, the term "executable module" or "executable component" can refer to software objects, routings, or methods that may be executed on the computing system. The different components, modules, engines, and services described herein may be implemented as objects or processes that execute on the computing system (e.g., as separate threads). Such executable modules may be managed code in the case of being executed in a managed environment in which type safety is enforced, and in which processes are allocated their own distinct memory objects. Such executable modules may also be unmanaged code in the case of executable modules being authored in native code such as C or C++.

[0017]   In the description that follows, embodiments are described with reference to acts that are performed by one or more computing systems. If such acts are implemented in software, one or more processors of the associated computing system that performs the act direct the operation of the computing system in response to having executed computer-executable instructions. For example, such computer-executable instructions may be embodied on one or more computer-readable media that form a computer program product. An example of such an operation involves the manipulation of data. The computer-executable instructions (and the manipulated data) may be stored in the memory

104 of the computing system 100. Computing system 100 may also contain communication channels 108 that allow the computing system 100 to communicate with other processors over, for example, network 110.

[0018]   Embodiments described herein may comprise or utilize a special purpose or general-purpose computer including computer hardware, such as, for example, one or more processors and system memory, as discussed in greater detail below. Embodiments described herein also include physical and other computer-readable media for carrying or storing computer-executable instructions and/or data structures. Such computer-readable media can be any available media that can be accessed by a general purpose or special purpose computer system. Computer-readable media that store computer-executable instructions are physical storage media. Computer-readable media that carry computer-executable instructions are transmission media. Thus, by way of example, and not limitation, embodiments of the invention can comprise at least two distinctly different kinds of computer-readable media: computer storage media and transmission media.

[0019]   Computer storage media includes RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other tangible storage medium which can be used to store desired program code means in the form of computer-executable instructions or data structures and which can be accessed by a general purpose or special purpose computer.

[0020]   A "network" is defined as one or more data links that enable the transport of electronic data between computer systems and/or modules and/or other electronic devices. When information is transferred or provided over a network or another communications connection (either hardwired, wireless, or a combination of hardwired or wireless) to a computer, the computer properly views the connection as a transmission medium. Transmissions media can include a network and/or data links which can be used to carry desired program code means in the form of computer-executable instructions or data structures and which can be accessed by a general purpose or special purpose computer. Combinations of the above should also be included within the scope of computer-readable media.

[0021]   Further, upon reaching various computer system components, program code means in the form of computer-executable instructions or data structures can be transferred automatically from transmission media to computer storage media (or vice versa). For example, computer-executable instructions or data structures received over a network or data link can be buffered in RAM within a network interface controller (e.g., a

"NIC"), and then eventually transferred to computer system RAM and/or to less volatile computer storage media at a computer system. Thus, it should be understood that computer storage media can be included in computer system components that also (or even primarily) utilize transmission media.

[0022]   Computer-executable instructions comprise, for example, instructions and data which, when executed at a processor, cause a general purpose computer, special purpose computer, or special purpose processing device to perform a certain function or group of functions. The computer executable instructions may be, for example, binaries, intermediate format instructions such as assembly language, or even source code. Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the described features or acts described above. Rather, the described features and acts are disclosed as example forms of implementing the claims.

[0023]   Those skilled in the art will appreciate that the invention may be practiced in network computing environments with many types of computer system configurations, including, personal computers, desktop computers, laptop computers, message processors, hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, mobile telephones, PDAs, pagers, routers, switches, and the like. The invention may also be practiced in distributed system environments where local and remote computer systems, which are linked (either by hardwired data links, wireless data links, or by a combination of hardwired and wireless data links) through a network, both perform tasks. In a distributed system environment, program modules may be located in both local and remote memory storage devices.

[0024]   Figure 2 schematically illustrates an example system 200 in which targeted recovery of an application may occur without recovering the entire system in accordance with the principles described herein. The system 200 includes an operational volume 210 accessible to multiple applications 201, so that the applications 201 may read to and write from the operational volume 210. For instance, the applications 201 are illustrated as including four applications 201A, 201B, 201C and 201D, although the ellipses 201E represent that the principles described herein are not limited at all to a system that operates a particular number of applications. The system 200 may operate a single application 201A or perhaps thousands of applications that all share the operational volume 210. A

"volume" is a single namespace that is recognized by the file system as including contiguous logical or physical addresses.

[0025] The operational volume 210 stores an application-specific data container 211A, although the operational volume 210 may store voluminous amounts of data other than the application-specific data container as represented by the ellipses 211B. The application-specific data container 211A is associated with application 201A as represented by the line 212. The application-specific data container 211A may be, for example, an application-specific file.

[0026] In one example, the applications 201 may each be virtual machines. In that case, the application 201A may be a virtual machine, and the application-specific data container 211A might be a virtual hard drive associated with the application 201A. In that case, the system 200 may be a portion of a host computing system that hosts multiple virtual machines. The host computing system may have multiple volumes, although only the one volume 210 is illustrated in Figure 2.

[0027] The principles described herein permit for convenient recovery of an application-specific data container without recovering the entire volume in which the application-specific data container is stored. For instance, in Figure 2, the application-specific data container 211A may be recovered without recovering the entire volume 210. This is helpful in that sometimes the application-specific data container 211A may become corrupted or untrusted, while the volume 210 itself remains uncorrupted as a whole. Furthermore, recovery of the entire volume might take much longer than the recovery of just a portion of the volume.

[0028] Significantly, the system 200 further operates such that the application 201A may continue to operate on the application-specific data container 211A while the recovery of the application-specific data container 211A is occurring in the background. Thus, from the perspective of the application 201A, the recovery happens quickly, and perhaps in an imperceptible amount of time, whilst the recovery actually occurs in the background over a much longer period of time.

[0029] In order to facilitate this targeted and rapid recovery, the system 200 includes a number of executable modules including a management module 221, a snapshot module 222, and a recovery module 223. Each of these modules will be further described below.

[0030] The modules 221 through 223 may each be instantiated and/or operated by a computing system (such as computing system 100) in response to the processors (e.g., processor(s) 102) of the computing system executing one or more computer-executable

instructions stored on a computing-readable storage media comprising a computer program product. Although specific acts are described as being associated with the modules 221 and 223, such acts need not be performed by the specific modules described.

[0031] In order to prepare for a possible need for recovery, the snapshot module 222 takes snapshots of at least the application-specific data container 211A but perhaps a larger segment of the volume 210 or even perhaps the entire volume 210 as a whole. These snapshots may be periodic and/or in response to particular events. In any case, Figure 2 illustrates an example in which the snapshot module has taken three snapshots 231A, 231B and 231C of the application-specific data container 211A and provided the snapshots 231A, 231B and 231C in a snapshot volume 230 as represented by multi-headed arrow 241. In this example, the snapshot 231A is the oldest snapshot, the snapshot 231B is the second snapshot taken, and the snapshot 231C is the most recent snapshot of the application-specific data container 211A.

[0032] The management module 221 determines if and when the application-specific data container is to be subject to recovery from a snapshot within the snapshot volume 230. In response, the recovery module 223 performs the recovery of the application-specific data container 211A from the snapshot 231C in the snapshot volume 230 to repopulate the content of the application-specific data container 221A as represented by the arrow 242. While the application-specific data container 211A might momentarily not be available to the application 201A while the recovery is initiated, immediately after the recovery is initiated, the management module 221 may permit the application 201A to perform read and write operations on the application-specific data container 211A.

[0033] Figure 3 illustrates a flowchart of a method 300 for recovering application specific data corresponding to an application. As the method 300 may be performed within the system 200 of Figure 2, the method 300 will now be described with respect to Figure 2.

[0034] The method 300 is initiated upon determining that a particular application-specific data container of an application is to be subject to recovery from a snapshot (act 301). For instance, referring to Figure 2, the management module 221 determines that the application-specific container 211A of the application 201A is to be subject to recovery from snapshot 231C.

[0035] In response, the method temporarily makes the application-specific data container unavailable to the application by, for example, closing the application-specific data container (act 302). For instance, in Figure 2, the management component 221 may

close the application-specific data container 211A. The application-specific data container 211A will remain closed for but a brief time compared to the total length of the recovery operation. Furthermore, the application-specific data container 211A is closed without closing the operational volume 210, thus keeping the operational volume 210 available to at least one other application (e.g., application 201B) while the application-specific data container 211A is closed.

[0036] The recovery operation is then initiated (act 303). For instance, in the case of Figure 2, the recovery module 223 initiates recovery by beginning to copy a prior state of content of the application-specific data container 201A from a prior snapshot 231C as represented by arrow 242. The recovery module 223 will proceed to copy from the prior snapshot 231C one region of the application-specific data container 201A at a time. Prior to recovering a region, the recovery module 223 will take a range lock for that region.

[0037] Once the recovery operation is initiated, the application-specific data container is opened (act 305) thereby permitting the application to perform read and write operations on the application-specific data container, so long as there are no range locks held by the recovery module on the region that contains the memory location being read from or written to. The application-specific data container is opened even though the recovery operation continues (act 304), and in fact, may have only begun moments before. Thus, referring to Figure 2, the recovery module 223 continues the recovery in the background (as represented by arrow 242), while the management module 221 continues to allow the application 201A to perform read and write operations on the application-specific data container 211A.

[0038] While the recovery is still in process (act 304), the management module 221 operates to respond to read and write requests from the application 201A in a manner that the read and write requests are honored consistent with the recovery as described herein.

[0039] For instance, upon detecting that the application 201A issues a read request for an address region of content from the application-specific data container ("Yes" in decision block 311), the management module 221 causes the recovery to momentarily pause for that address region (act 312) such that if the recovery approaches close to that address region, or even reaches that address region, the recovery pauses. The read operation is performed (act 313), and thereafter the recovery for that address region is resumes such that if the recovery were to approach or arrive at the region, the recovery would not pause (act 314). As an example, the read operation may be performed in the manner illustrated and described with respect to Figure 4. The acts 312 through 314 might

be performed each time the application 201A issues a read request to the application-specific data container 211A.

[0040]   Upon detecting that the application 201A issues a write request to an address region of the application-specific data container ("Yes" in decision block 321), the management module 221 again causes the recovery to momentarily pause for that address region (act 322) such that if the recovery approaches close to that address region, or even reaches that address region, the recovery pauses. The write operation is performed (act 323), and thereafter the recovery is resumes such that if the recovery were to approach or arrive at the region, the recovery would not pause (act 324). As an example, the write operation may be performed in the manner illustrated and described with respect to Figure 5. The acts 312 through 314 might be performed each time the application 201A issues a read request to the application-specific data container 211A.

[0041]   Figure 4 illustrates a method 400 for responding to a read request while the recovery operation is still in process. The method 400 may be performed in order to perform the read operation of act 313 in Figure 3. Thus, the method 400 may be initiated after act 312 in Figure 3.

[0042]   The method 400 determines whether the address region corresponding to the read request is already restored to the application-specific data container in the operational volume (decision block 401). This may be accomplished by the recovery module 223 consulting a recovery progress status 251. For instance, if the recovery module 223 copies the content from the snapshot 231C sequentially, the recovery progress status 251 might represent a high water mark representing the highest address recovered into the application-specific data container. In addition, this may be accomplished by the recovery module 223 consulting an already written indication 252 that the address region of the application-specific data container has been written to since the beginning of the recovery. For instance, the address region might have been written to in response to the performance of method 500 or in response to act 406 in Figure 4 described further below, even though the address region has not yet been reached yet through the normal recovery performed by act 304.

[0043]   If the address region for the read request is already restored to the application-specific data container ("Yes" in decision block 401), the content of the address region is read from the application-specific data container in the operational volume (act 402) and is thus made available to the application 201A in response to the read request (act 403). If the address region for the read request is not already restored to the application-specific

data container ("No" in decision block 401), the content is read instead from the corresponding portion of the snapshot (act 404), and the content is provided to the application (act 403) whilst the read content is also optionally provided to the application-specific data container (act 405). Furthermore, the already written indication is optionally updated (act 406) to reflect that a write has occurred to the address region of the application-specific data container since the recovery began. Once the read operation completes, the method 400 returns to act 314 of Figure 3, where the recovery operation resumes. The method 400 may be repeated numerous times for each read request received during the recovery process. Once the recovery process completes (act 306), read operations may continue as normal, without intervention from the management module 221.

[0044] Figure 5 illustrates a method 500 for responding to a write request while the recovery operation is still in process. The method 500 may be performed in order to perform the write operation of act 323 in Figure 3. Thus, the method 500 may be initiated after act 322 in Figure 3.

[0045] The method 500 performs the write operation by writing the associated content of the write request is written to the address region of the application-specific data container in the operation volume (act 501). The method 500 also determines whether the address region corresponding to the write request is already restored to the application-specific data container in the operational volume (decision block 502). This again may be accomplished by the recovery module 223 consulting a recovery progress status 251. In addition, this may be accomplished by the recovery module 223 consulting an already-written indication 252 that the address region of the application-specific data container has been written to since the beginning of the recovery. For instance, the address region might have been written to in response to a previous performance of method 500 or in response to act 406 in Figure 4, even though the address region has not yet been reached yet through the normal recovery performed by act 304.

[0046] If the address region for the write request is already restored to the application-specific data container ("Yes" in decision block 502), this completes the write operation, thereby returning to act 324 in Figure 3 to resume the restore operation. If the address region for the write request has not already been restored to the application-specific data container ("No" in decision block 502), the already written indicator 252 is updated to reflect that the address region has been written to since the recovery operation began (act 503). Thereafter, again thereby returning to act 324 in Figure 3 to resume the restore

operation. The method 500 may be repeated numerous times for each write request received during the recovery process. Once the recovery process completes (act 306), write operations may continue as normal, without intervention from the management module 221.

[0047] From the perspective of the recovery module, the recovery module 223 determines that a particular application-specific data container of an application is to be subject to recovery from a snapshot. For instance, the recovery module 223 may have received an instruction to recover from the management module 221 in response to the management module 221 performing act 301 in Figure 3. In response, the recovery module initiates recovery.

[0048] The recovery module continues the recovery by copying content from the snapshot to the application-specific data container one portion at a time, which is analogous to act 304 in Figure 3. The recovery module does respond to instructions to pause and resume recovery in response to acts 312, 314, 322 and 324, but otherwise simply moves forward with recovery. Despite such momentary pauses, the recovery operation will still be said to be continuing as shown in act 304.

[0049] The recovery module also tracks at least one address region corresponding to a write request that is made on the application-specific container in the operational volume apart from the recovery. For instance, this may be accomplished using the already-written indicator 252 in Figure 2. When copying one portion at a time from the snapshot, the recovery module skips copying for any addresses that have already been written as indicated within the indicator 252. This prevents the recovery from overwriting newer data created by allowing the application 201A to continue processing during the recovery.

[0050] An example implementation of how collision avoidance may be facilitated will now be described, first from the perspective of the read/write operation, and then from the perspective of the recovery.

[0051] When the application issues a read or write request, the system takes a range lock for the region addressed by the read or write request. If the recovery is working on the same range, then this read/write operation will be pended to be resumed later. Second, a high water mark of the recovery (an example of the recovery progress status 251) is evaluated, along with a dirty map (an example of the already-written indicator 252) is evaluated, to determine the appropriate action as described above. For instance, for a read operation, it is determined whether to read from the data container (act 402), or whether to read from the snapshot (act 403). For a write operation, it is determined whether the dirty

map needs to be updated to reflect a write (act 503). The appropriate I/O operation is then performed, and then the range lock is released for the region. If a recovery got pended for this region, then the recovery is resumed for the region.

[0052] The collision avoidance from the recovery perspective occurs one range of addresses at a time. For each region to be recovered, the recovery process takes a range lock for the region to be recovered. If a read/write operation already has a range lock on that region, the recovery on the region is pended to be resumed later on read/write completion. Once the recovery is able to obtain the range lock on that region, recovery is performed based on the dirty map. For instance, recovery will skip copying from the snapshot for all of those addresses that are marked as already having been written to in the dirty map. Once the region is recovered, the recovery process release the range lock. This allows any pended read/write request to complete on that range.

[0053] Thus, the principles described herein provide an effective and efficient mechanism for targeted recovery of an application-specific data container without requiring recovery of the entire volume, and also while allowing the application to continue operating on the application-specific data container that is in the process of being recovered.

[0054] The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

CLAIMS

1.   A method for recovering application specific data corresponding to an application, the method comprising:

an act of determining that a particular application-specific data container of an application is to be subject to recovery from a snapshot;

in response to the act of determining, an act of initiating recovery of a prior state of content of the application-specific data container from a prior snapshot by beginning to copy a prior version of the content from a snapshot to the application-specific data container in an operation volume accessible by the application; and

while the content of the application-specific data container is still being copied from the snapshot to the application-specific data container, an act of permitting the application to perform read and write operations on the application-specific data container.

2.   The method in accordance with Claim 1, further comprising:

after the act of permitting the application to perform read and write operations on the application-specific container, but before the recovery is complete, an act of detecting that the application issued a read request for an address region of content from the application-specific data container.

3.   The method in accordance with Claim 2, further comprising:

an act of pausing the recovery after the an act of detecting that the application issued a read request for an address region of content from the application-specific data container;

an act of performing the read operation; and

after the act of performing the read operation, an act of resuming the recovery.

4.   The method in accordance with Claim 2, further comprising the following in response to the read request:

an act of determining that the address region corresponding to the read request is already restored to the application-specific data container in the operational volume; and

in response to the act of determining that the address region corresponding to the read request is already restored, an act of reading the content of the address region from the application-specific data container in the operation volume.

5.   The method in accordance with Claim 2, further comprising the following in response to the read request:

an act of determining that the address region corresponding to the read request has not already been restored to the application-specific data container in the operational volume;

in response to the act of determining that the address region corresponding to the read request has not already been restored, an act of reading the content from the snapshot; and

an act of providing the read content to the application.

6.   The method in accordance with Claim 1, further comprising:

after the act of permitting the application to perform read and write operations on the application-specific container, but before the recovery is complete, an act of detecting that the application issued a write request to an address region of the application-specific data container.

7.   The method in accordance with Claim 6, further comprising:

an act of pausing the recovery after the an act of detecting that the application issued a write request to an address region of the application-specific data container;

an act of performing the write operation; and

after the act of performing the write operation, an act of resuming the recovery.

8.   The method in accordance with Claim 1, further comprising the following after the act of determining that the application-specific data container of an application is to be subject to recovery from a snapshot and before the act of initiating recovery:

an act of closing the application-specific data container so as to not be accessible to the application.

9.   The method in accordance with Claim 1, wherein the application comprises a virtual machine.

10.   A system in which targeted recovery of an application may occur without recovering the entire system, the system comprising:

an operational volume accessible to a plurality of applications, the operational volume including an application-specific data container corresponding to a particular application of the plurality of applications;

a snapshot module configured to take snapshots of at least the application-specific container;

a management module configured to determining that application-specific data container is to be subject to recovery from a snapshot taken by the snapshot module; and

a recovery module configured to perform recovery of the application-specific data container when determined by the management module, wherein the management module; permits the application to perform read and write operations on the application-specific data container while the recovery module is performing recovery.
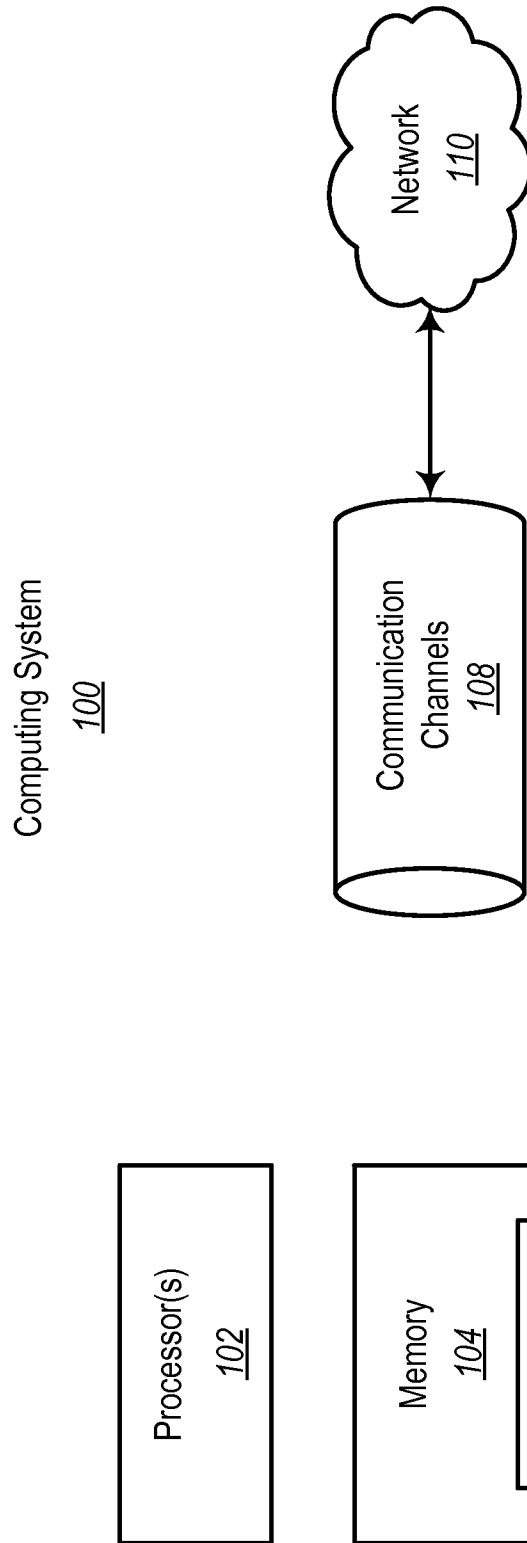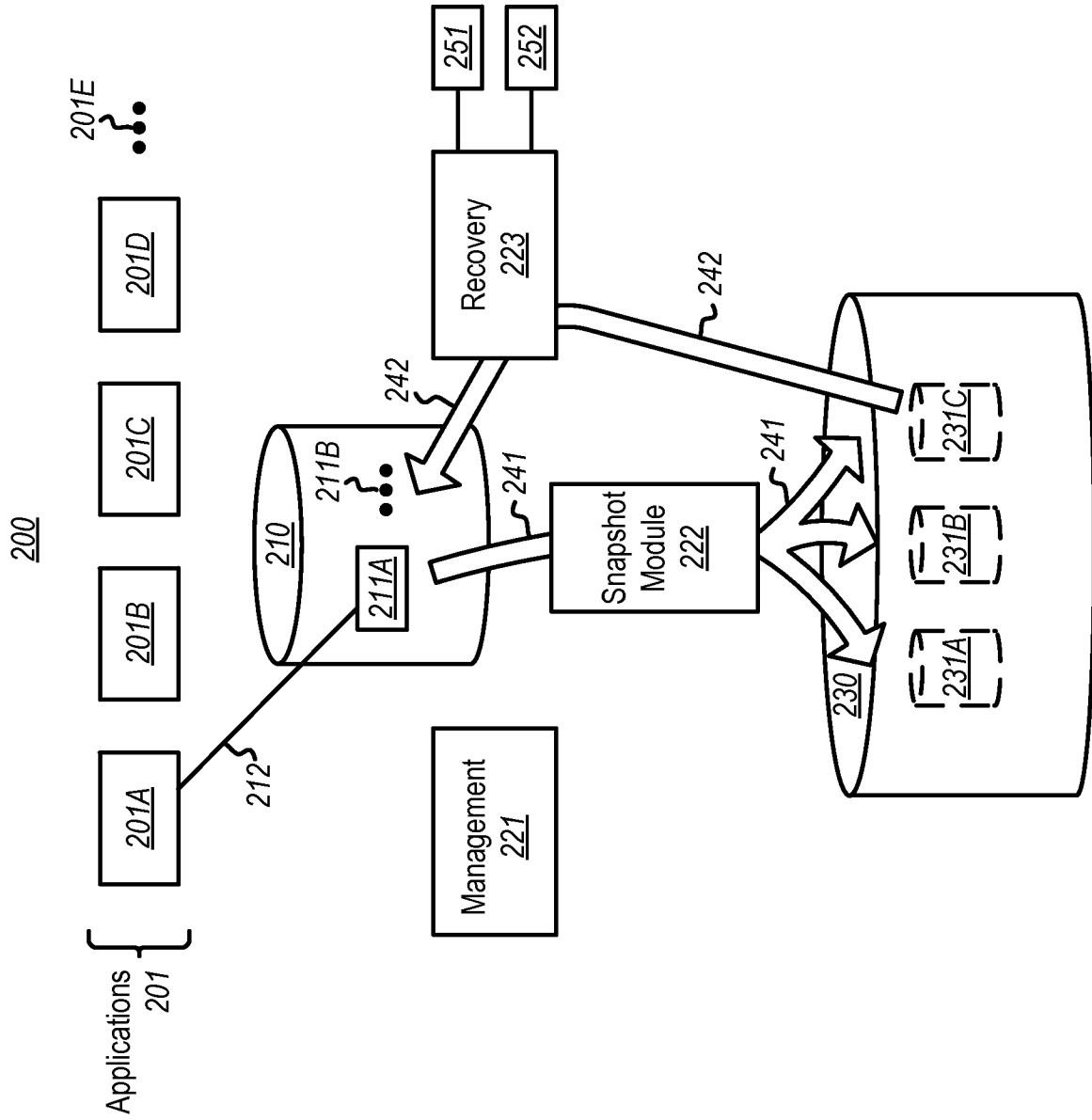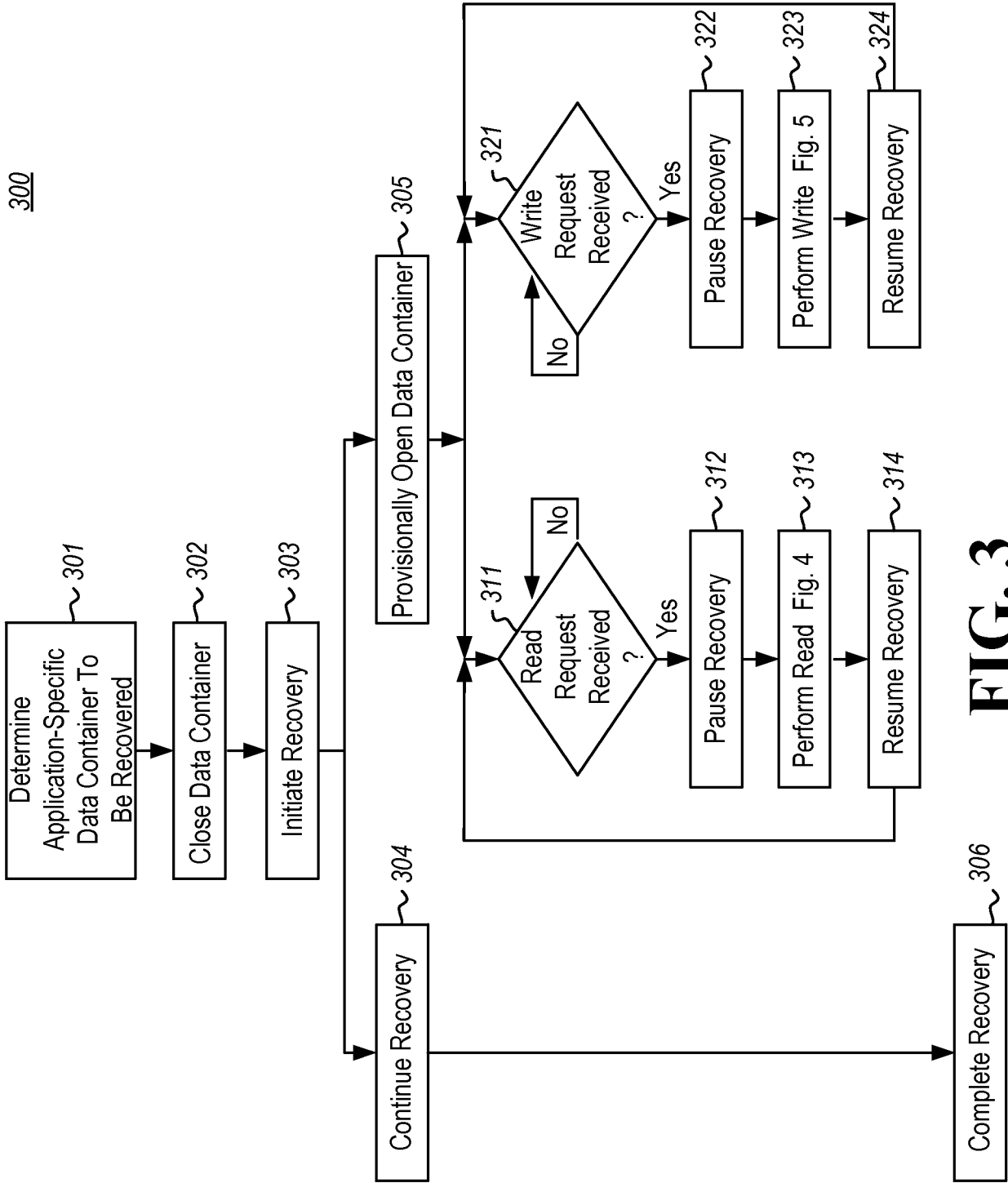
FIG. 1

FIG. 2
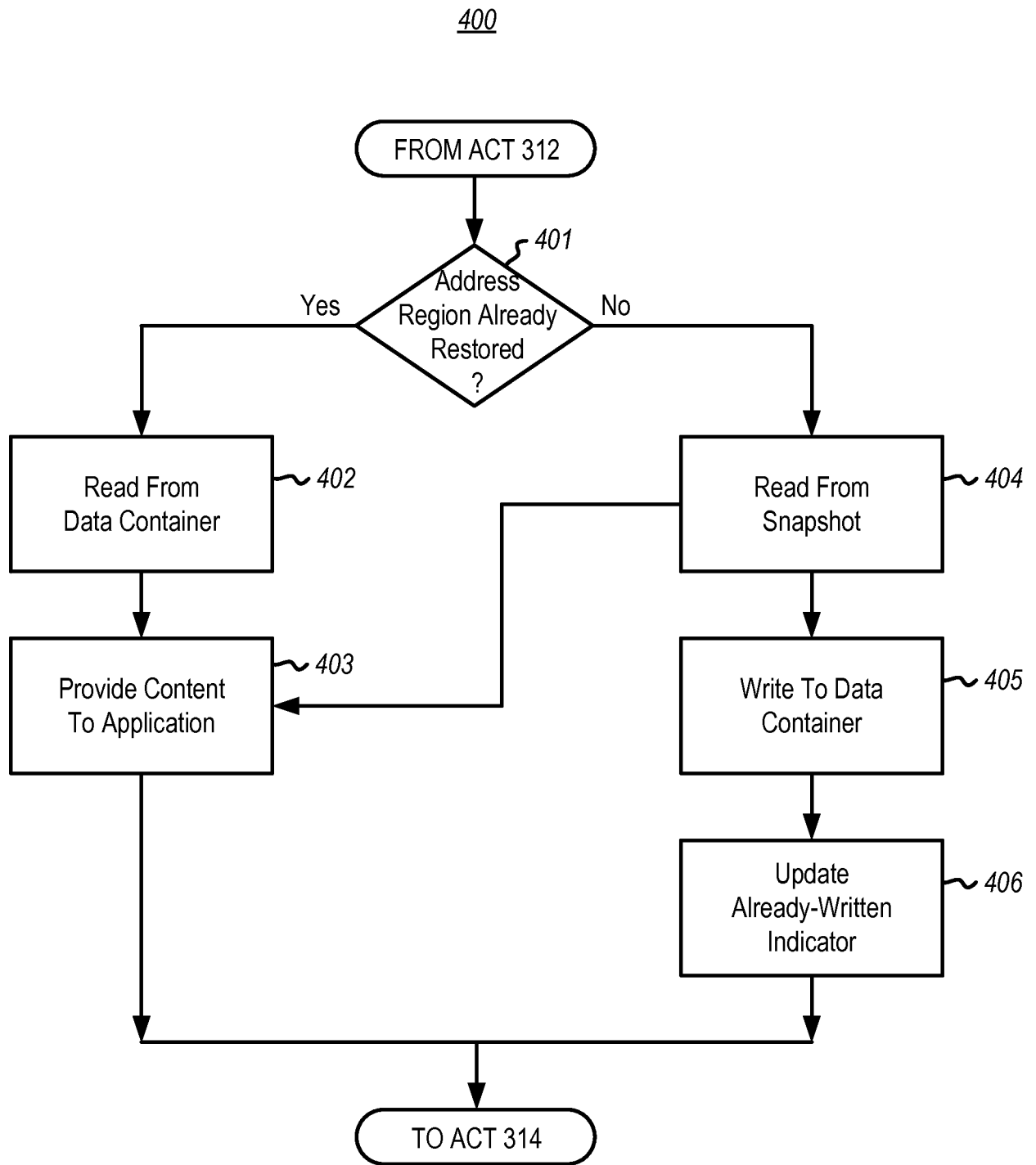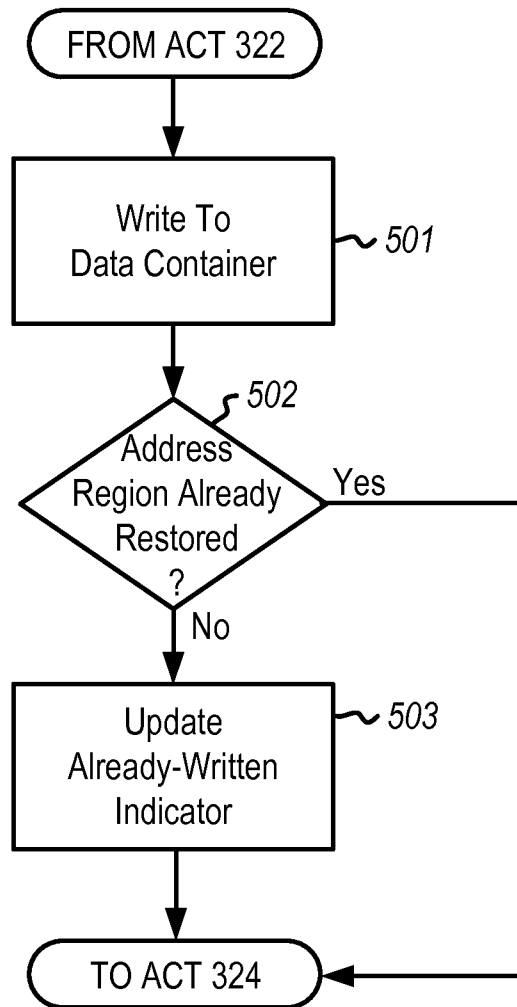
FIG. 3

_400_

FIG. 4

_500_

```
        ┌─────────────────────┐
        │    FROM ACT 322     │
        └─────────────────────┘
                   │
                   ▼
        ┌─────────────────────┐
        │      Write To       │  ～ 501
        │   Data Container    │
        └─────────────────────┘
                   │
                   ▼  ╭ 502
                  ╱◇╲
                 ╱     ╲        Yes
            ┌─  Address  ─┐ ──────────────┐
             ╲ Region Already╱             │
              ╲ Restored   ╱               │
               ╲    ?    ╱                 │
                ╲  ◇  ╱                     │
                   │ No                    │
                   ▼                       │
        ┌─────────────────────┐            │
        │      Update         │  ～ 503     │
        │  Already-Written    │            │
        │     Indicator       │            │
        └─────────────────────┘            │
                   │                       │
                   ▼                       │
        ┌─────────────────────┐            │
        │     TO ACT 324      │◄───────────┘
        └─────────────────────┘
```

FIG. 5

INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F11/14
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2007/055833 A1 (VU NGOCLAN T [US] ET AL VU NGOCLAN THI [US] ET AL) 8 March 2007 (2007-03-08)	1-7,9,10
Y	paragraph [0009] paragraph [0053] - paragraph [0055] -----	8,9
X	US 7 822 758 B1 (PRAKASH ASHISH [US] ET AL) 26 October 2010 (2010-10-26)	1,9,10
Y	column 2, line 65 - column 3, line 33 -----	8,9
X	US 2012/017114 A1 (TIMASHEV RATMIR [CH] ET AL) 19 January 2012 (2012-01-19)	1,9,10
Y	paragraph [0016] paragraph [0038] - paragraph [0044] paragraph [0051] - paragraph [0053] paragraph [0064] figures 1-4 abstract; claim 1 -----	8,9
	-/--	

[X] Further documents are listed in the continuation of Box C.   [X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
17 July 2014	24/07/2014

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Lanchès, Philippe

1

C(Continuation).   DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2012/151136 A1 (HAY GADI [IL] ET AL) 14 June 2012 (2012-06-14)	1-7,10
Y	paragraph [0016] - paragraph [0017] paragraph [0022] paragraph [0028] -----	8,9
X	US 2004/030951 A1 (ARMANGAU PHILIPPE [US]) 12 February 2004 (2004-02-12)	1-7,9,10
Y	paragraph [0096] - paragraph [0106] -----	8,9
X	US 7 702 868 B1 (HARDMAN DANIEL H [US]) 20 April 2010 (2010-04-20)	1-7,9,10
Y	figure 4 column 2, lines 35-59 column 4, lines 39-67 -----	8,9

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
US 2007055833	A1	08-03-2007	CN	101258472 A	03-09-2008
			EP	1922624 A2	21-05-2008
			HK	1121839 A1	26-08-2011
			US	2007055833 A1	08-03-2007
			WO	2007030304 A2	15-03-2007
US 7822758	B1	26-10-2010	NONE		
US 2012017114	A1	19-01-2012	EP	2596426 A2	29-05-2013
			US	2012017114 A1	19-01-2012
			US	2012284236 A1	08-11-2012
			US	2014047268 A1	13-02-2014
			WO	2012012365 A2	26-01-2012
US 2012151136	A1	14-06-2012	US	2012151136 A1	14-06-2012
			US	2012254535 A1	04-10-2012
US 2004030951	A1	12-02-2004	NONE		
US 7702868	B1	20-04-2010	US	7398365 B1	08-07-2008
			US	7702868 B1	20-04-2010