Title: APPLICATION CONFIGURATION IN DISTRIBUTED STORAGE SYSTEMS

Abstract: A method and system for enabling dynamic matching of storage utilization characteristics of a host system application with the characteristics of the available storage pools of an attached distributed storage system, in order to provide an optimal match between the application and selected storage pool. An abstraction manager is provided, enhanced with a storage device configuration utility/module, which performs a series of tasks to (1) obtain/collect the correct configuration information from each connected storage device or storage pools and/or (2) calculate the configuration information when the information is not readily available. The storage device configuration module then normalizes, collates and matches the configuration information to the various applications running on the host system and/or outputs the information to a user/administrator of the host system via a software interface. By using the collected/calculated configuration information, applications are assigned to an optimal storage pool in an intelligent and efficient manner.
Published:
— with international search report
— before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments

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.
APPLICATION CONFIGURATION IN DISTRIBUTED
STORAGE SYSTEMS

Technical Field

The present invention relates generally to computer systems and in particular to distributed storage systems. Still more particularly, the present invention relates to a method and system for dynamically configuring distributed storage systems.

Background of the Invention

Over the last several years, significant changes have occurred in how persistent storage devices are attached to computer systems. With the introduction of Storage Area Network (SAN) and Network Attached Storage (NAS) technologies, storage devices have evolved from locally attached, low capability, passive devices to remotely attached, high capability (intelligent), active devices that are capable of deploying vast file systems and file sets. The remotely-attached intelligent storage devices are referred to herein as "storage servers," while the computer system to which they are attached are referred to as "hosts." These devices are interconnected via a network interface and collectively referred to as distributed storage systems.

One complication present within a distributed storage environment is that applications which execute on the host, particularly databases and file systems, are not cognizant of the type of storage that they are utilizing. Hosts typically have multiple storage connections. For example, a single host may be connected to one or more storage servers and one or more locally-attached disk drives. Then, a system administrator on the host creates virtual storage pools from among these storage connections. The administrator may create one pool comprising storage from a locally attached disk and storage from one of the storage servers, another pool comprising storage from a different locally attached disk, and yet another pool comprising two distributed storage servers. The administrator then assigns applications to the respective storage pools.

One problem with this approach is that, in order to complete the assignment in an optimal manner, the administrator has to match the storage utilization characteristics of the application with the
characteristics of the pool. To complete this match, certain configuration information on the various storage pools needs to be obtained. However, with conventional implementations, the configuration information is generally not obtainable for even a single storage connection, and even more unavailable for a virtualized storage pool.

SUMMARY OF THE INVENTION

Disclosed is a method and system for enabling dynamic matching of storage utilization characteristics of a host system application with the characteristics of the available storage pools of an attached distributed storage system, in order to provide an optimal match between the application and selected storage pool. An abstraction manager is provided, enhanced with a storage device configuration utility/module, which performs a series of tasks to (1) obtain/collection the correct configuration information from each connected storage device or storage pools and/or (2) calculate the configuration information when the information is not readily available. The storage device configuration module then normalizes, collates and matches the configuration information to the various applications running on the host system and/or outputs the information to a user/administrator of the host system via a software interface. By using the collected/calculated configuration information, applications are assigned to an optimal storage pool in an intelligent and efficient manner.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the present invention will now be described, by way of example only, with reference to the following drawings:

Figure 1 is a diagram of a distributed network having distributed storage connected to host systems, within which embodiments of the invention may advantageously be implemented;

Figure 2 is a block diagram of an exemplary host system with software utility/modules for performing the configuration data retrieval from distributed storage, according to one embodiment of the invention; and

Figure 3 is a flow diagram illustrating the various process of collecting and/or calculating, then collating configuration data via the
device configuration utility of an abstraction manager in accordance with one embodiment of the invention.

DETAILED DESCRIPTION OF AN ILLUSTRATIVE EMBODIMENT

Embodiments of the present invention provides a method and system for enabling dynamic matching of storage utilization characteristics of a host system application with the characteristics of the available storage pools of a distributed storage system, in order to provide an optimal match between the application and selected storage pool. An abstraction manager (described below) is enhanced with a storage device configuration utility/module, which performs a series of tasks, including obtaining/collnecting the correct configuration information from each connected storage device (or storage pools) and/or calculating the configuration information when the information is not readily available. The storage device configuration module then normalizes, collates and matches the configuration information to the various applications running on the host system and/or outputs the information to a user of the host system via a software interface. Using the collected/calculated configuration information, applications may then be assigned (manually by the user or automatically by the abstraction manager) to the correct storage pool, in an intelligent and efficient manner. The abstraction manager and device configuration module are described in greater details below.

Implementation of embodiments of the present invention occur within a distributed storage environment that includes both local and remote storage according to the illustrative embodiment. Of course, it is understood that the environment may comprise only one type of storage, e.g., remote storage, and the functional features described below apply regardless of the actual configuration of the storage environment. Further, specific implementation of embodiments of the present invention are carried out within a host system in the described embodiment. An exemplary distributed storage environment and host system are now described to provide a context within which the inventive features may advantageously be implemented.

Turning now to the Figures, Figure 1 illustrates an exemplary embodiment of the topology of a distributed storage system, within which the various features may advantageously be implemented. As shown by Figure 1, distributed storage system comprises one or more host systems (for example, host systems 101 and 102) connected to one or more storage
servers (for example, servers 105 and 106) via a first internal/external network 103. Storage servers 105/106 are themselves connected to persistent storage devices (disks) 109 via a second internal/external network 107. Both first network 103 and second network 107 comprise some combination of fiber channels or Ethernet or other network structure based on system design.

While Figure 1 illustrates only two hosts (101 and 102) connected to two storage servers (105 and 106) using fiber channel, it is understood that any number of host systems and/or storage systems may exist within the distributed storage system. Also, while storage servers 105/106 are themselves connected to eight disks (or persistent storage devices) via another fiber channel network, the number of disks is variable and not limited by the illustration. Finally, embodiments of the invention can be independent of the physical network media connecting the components. For example, all of the fiber channel networks could be replaced with Ethernet networks or other network connection medium.

With reference now to Figure 2, there is illustrated a block diagram representation of an exemplary host system, which for illustration is assumed to be host system 101. Host system 101 is a typical computer system that includes a processor 201 connected to local memory 203 via a system bus 202. Within local memory 203 are software components, namely operating system (OS) 205 and application programs 207. Host system 101 also includes the required hardware (e.g., network interface device, NID 209) and software components to enable connection to and communication with a distributed storage network.

To enable access to a distributed storage network, applications 207, such as databases and file systems (not specifically shown), execute on the host system 101 accessing virtualized storage pools (not shown). These storage pools are constructed by the host system(s) using file systems 215 and/or logical volume managers 213 (associated with or component parts of the OS 205) and are physically backed by actual storage residing at one or more of the storage servers or at local storage 219, directly attached to the host system 101. It is understood that some operating systems do not have an explicit volume manager but rather combine that function with the file system. The illustrative embodiment provides LVM 213, but the approaches are generally applicable to either configuration.
As applications issue input/output (I/O) operations to the storage pools, these requests are passed through the host file system 215, host logical volume manager 213, and host device drivers 214. These requests are then converted to an appropriate form and transmitted to the appropriate locally-attached or remote storage device. This described processing pipeline (i.e., host file system, host logical volume manager, host device driver) along with storage network protocol and storage server modules are collectively referred to as the distributed storage system software stack.

According to one embodiment of the invention, a particular module is provided in each host system's storage software stack to interface directly with each physical storage device in the storage pool and present a logical contiguous view of the storage pool to applications running on the host system. In some embodiments, this module is a provided within (or associated with) the host logical volume manager, while in other systems that do not have logical volume managers, the functionality provided by the module is integrated into the host's file system. Regardless of the implementation, this module is generally (or collectively) referred to as the abstraction manager.

As illustrated by Figure 2, host system 101 includes abstraction manager 211, which serves as the host's system's point of data collection and configuration for the entire distributed storage. While illustrated as a separate module (block), abstraction manager 211 may be a component of or associated with OS 125 and specifically associated with FS 215 or with LVM 213, where an explicit LVM is provided.

According to the described embodiment, abstraction manager 211 is the one host component that has both (1) knowledge of the physical device composition of the pools and (2) ability to interact with the software modules controlling those devices. Abstraction manager 211 is therefore the one point in the stack that knows the physical composition of the storage pools which the applications interface with, and abstraction manager 211 is thus the module that is able to efficiently collect and calculate the required data.

The abstraction manager interfaces with each storage device via respective device drivers. The abstraction manager converts references from logical partitions into the corresponding actual physical devices. Within the described embodiments, the primary functions of the abstraction
manager are gathering, collecting and collating the configuration information and presenting the information to the applications and host system administrator/user. Notably, the abstraction manager does not itself utilize the retrieved or calculated configuration information. Rather, the abstraction manager simply collects and calculates the data, which reflects the characteristics of a particular storage pool. Applications and administrators may then utilize this data to match the characteristics of the data to the requirements of a particular application. That is, the applications and administrators utilize the data to determine the best storage pool (from among all available pools) to use for the particular application.

To complete the above referenced functions of the described embodiment of the invention, the abstraction manager is enhanced with an additional module that collects, calculates and distributes the requisite configuration information of the various storage devices. For ease of description herein, that module is referred to as the storage device configuration (SDC) module. As described in greater details below, the SDC module also provides a software interface, which is utilized by both the applications and the user(s) to match storage pools with specific applications and vice versa. Thus, as an example, applications may query the interface to automatically match functions to the appropriate storage pool and/or the information can be presented to the system administrators for manually matching functions to storage pools.

Figure 2 illustrates abstraction manager 211 having therein storage device configuration (SDC) utility/module 213 which, along with the other software components executing on the host system, enables the various functional features. SDC module 213 performs two primary functions: (1) SDC module 213 generates queries that enable retrieval of configuration information from the storage devices within the distributed storage system; and (2) SDC module 213 calculates and receives input of other data that is not collected from the storage device.

The configuration information is then utilized in determining which storage device to allocate to specific application processes. Examples of configuration information obtained about the storage pools by SDC module 213 and utilized to match the storage pools to specific applications (or applications to storage pools) are the following: the maximum transfer size that the pool can handle in any single operation; the average total throughput (MB/sec) that the pool can sustain; and the average latency
(e.g., in seconds) that can be expected from a single I/O operation on the pool.

The illustrative embodiment of the invention provides a method and/or system by which applications are able to obtain this data and utilize this data to configure the particular application for optimal use of the storage pools without the intervention of an administrator.

Implementation of embodiments of the invention includes providing the functionality of the SDC module into the abstraction manager to enable the above and other configuration information to be made available programmatically to the applications.

The processing completed by the abstraction manager 211 (particularly by the SDC module 213) includes three main operations, namely, data collection, data calculation, and data distribution. Each operation is now presented/described in delineated sections below, loosely following the process flow chart of Figure 3.

Data Collection

Data collection is the primary functions provided by the SDC utility and is automatically initiated once the storage device is connected to the host system, either directly (local storage) or via the network (remote storage). Thus, the process begins at block 302 with the connection of a storage device to the host system. SDC utility generates a query (block 304) to retrieve particular configuration data from the storage device. Notably, while it is anticipated that some information is available for each storage device, the described embodiment of the invention is applicable to configurations in which not all of the configuration information is obtainable even on a single storage device basis. Thus, decision block 306 provides a determination whether the storage device has its configuration information readily available. Assuming that some of the configuration information is obtainable, the SDC utility implements a process of retrieval/collection of the configuration data from the various storage devices, as shown at block 308.

In certain implementations, some information may exist on the storage device but not readily obtainable by a general query. For example, the original equipment manufacturer (or vendor) may configure the storage device with unique configuration data. To address the situations where
the information is not directly obtainable from the storage device with a general query, the abstraction manager is further enhanced with device-specific modules for the collection and translation of these unique configuration data.

One function of these device-specific modules is to provide these unique configuration data in a normalized format for use within data calculation and other functions of SDC utility. The data which the SDC utility wishes to collect from the physical devices may not be available and/or may be in a different form than the data collected from other physical devices. For example, a first device may report average latency in .001 (thousandths) of a second, a second device reports average latency in .0001 (ten-thousandths) of a second, while a third device does not report average latency at all. Given this scenario, device specific modules are created within the SDC utility that recognize, for example, that the desired method of reporting latency is in .0001 of a second. The device-specific module for the first device thus converts the latency numbers of the first device to .0001 of a second. The device-specific module for the second device does nothing since it is already reporting average latency in the correct norm/format, while the device-specific module for the third device begins timing actual I/Os in .0001 of a second so normalized latency numbers may be computed.

For simplicity in the description of the inventive methods, the combination of these device-specific modules and general SDC module, described above, are collectively referred to as SDC utility, which encompasses a generally applicable query/retrieval function across substantially all storage devices.

Data calculations

SDC utility provides two types of data calculations. The first calculation involves translating information collected from each individual storage device into data applicable to the entire pool (block 310). For example, the maximum transfer size for a pool is calculated to be the minimum of the maximum transfer sizes over all the individual storage devices. Since all of the data reported has to be representative of the pool and not the individual device, all of the data has to go through this calculation step.

The second calculation involves generating data that is not obtainable from an individual storage device or is difficult to extrapolate to a pool
basis (step 312). This second calculation is needed if the storage device does not report its configuration information. With this second calculation, the SDC utility abstracts the physical devices into the pool. Then, the abstraction manager translates all application I/O requests directed to the pool into the physical device components of the request. Finally, the abstraction manager forwards the appropriate request to each individual storage device.

One example of data that is generated via the second calculation is average latency for a single I/O operation to the pool. According to one implementation, abstraction manager is modified to statistically time and track I/O requests to the individual storage devices in order to compute average I/O latencies, average sustained I/O throughput to the pool, and other data associated with the pool and individual devices corresponding thereto.

Data Distribution via Software Interface

Data distribution is provided following data collection and data calculation (block 316). According to the described embodiment, data distribution is completed via a particular software interface within SDC utility that is constructed for the purpose of querying the abstraction manager to obtain the collected and calculated configuration data for a selected storage pool. This interface is accessible to all host system applications with the appropriate permissions, and enables applications to collect the configuration information. Applications are able to query this interface to automatically match functions to the appropriate storage pool.

Additionally, in one embodiment, the configuration information may be presented to the system administrator/user for manually matching functions to storage pools. This latter embodiment involves an output mechanism by which the configuration information is outputted on an output device of the host system once the information is collected, analyzed and translated. The administrator/user then manually selects the storage pools and links the pools to the appropriate applications (and vice versa).

It is important that while an illustrative embodiment of the present invention has been, and will continue to be, described in the context of a fully functional computer system with installed management software, those skilled in the art will appreciate that the software aspects of an
illustrative embodiment of the present invention are capable of being distributed as a program product in a variety of forms, and that an illustrative embodiment of the present invention applies equally regardless of the particular type of signal bearing media used to actually carry out the distribution. Examples of signal bearing media include recordable type media such as floppy disks, hard disk drives, CD ROMs, and transmission type media such as digital and analogue communication links.
CLAIMS

1. In a distributed storage system, a method comprising:
   querying one or more storage devices connect to a host system for
   configuration information of the one or more storage devices;
   responsive to receipt of the configuration information,
   automatically assigning specific ones of the one or more storage devices
   to an application running on the host system, based on said configuration
   information.

2. The method of Claim 1, further comprising:
   analyzing the configuration information received from the one or
   more storage devices to determine an optimal allocation of storage devices
   to the application; and
   wherein said automatically assigning further selects specific ones
   of the one or more storage devices that provide said optimal allocation.

3. The method of Claim 1, further comprising:
   grouping said one or more storage devices into pools;
   maintaining, via a management utility of the host component, a
   knowledge base of (1) a physical device composition of each pool of
   storage devices and (2) interaction with software modules controlling the
   storage devices; and
   collecting, calculating and distributing requisite configuration
   information of each storage device via a device configuration module of
   said management utility.

4. The method of Claim 3, further comprising:
   automatically initiating the collection of obtainable configuration
   information when the storage device is opened to collect configuration
   data of the storage devices; and
   when a first device provides at least one of different formats and
   recording of configuration data in a particular class from a second
   device, normalizing said configuration data into a pre-established format
   for recording said configuration data; wherein, said configuration module
   is enhanced with device specific modules for the collection and
   translation of device specific data.

5. The method of Claim 3, further comprising performing one of two
   types of data calculations from among:
translating information collected from each individual storage device into data applicable to the entire pool; and

generating data that is not obtainable from an individual storage device and data that is difficult to extrapolate to a pool basis.

6. The method of Claim 4, wherein said normalizing includes allocating the minimum of the maximum transfer sizes over all the individual storage devices as a maximum transfer size for the pool.

7. The method of Claim 6, further comprising:

translating all application I/O requests directed to the pool into its physical device components; and

forwarding the appropriate request to each individual storage device.

8. The method of Claim 1, further comprising:

statistically timing and tracking I/O requests to the individual storage devices in order to compute storage device characteristics, including average I/O latencies and average sustained I/O throughput to the pool; and

querying the manager to obtain the collected and calculated configuration data for a selected storage pool.

9. The method of Claim 8, further comprising:

providing an interface for access by applications and system administrators to the configuration data, such that (i) each application is able to query the interface to automatically match functions to the appropriate storage pool;

presenting the information to the system administrators for manually matching functions to storage pools; and

automatically assigning permissions to the interface to enable said interface to be accessible to all host system applications with said appropriate permissions.

10. A distributed storage system comprising:

at least one host system;

one or more storage devices connected to the at least one host systems and accessible via one or more host storage pools;

a software module that when executed on the at least one host system provides a plurality of functions form among:
querying one or more storage devices connect to a host system for
configuration information of the one or more storage devices;
collecting device configuration information for each host storage
pools within the distributed storage system; and
responsive to receipt of the configuration information,
automatically assigning specific ones of the one or more storage devices
to an application running on the host system, based on said configuration
information.

analyzing the configuration information received from the one or
more storage devices to determine an optimal allocation of storage devices
to the application; and

wherein said automatically assigning further selects specific ones
of the one or more storage devices that provide said optimal allocation.
grouping said one or more storage devices into pools;
maintaining, via a management utility of the host component, a
knowledge base of (1) a physical device composition of each pool of
storage devices and (2) interaction with software modules controlling the
storage devices;
collecting, calculating and distributing requisite configuration
information of each storage device via a device configuration module of
said management utility;
collecting configuration data of the storage devices by
automatically initiating the collection of obtainable configuration
information when the storage device is opened; and

when a first device provides at least one of different formats and
recording of configuration data in a particular class from a second
device, normalizing said configuration data into a pre-established format
for recording said configuration data; wherein, said configuration module
is enhanced with device specific modules for the collection and
translation of device specific data, wherein said normalizing includes
allocating the minimum of the maximum transfer sizes over all the
individual storage devices as a maximum transfer size for the pool.

11. The distributed storage system of Claim 10, further comprising:
one or more storage servers connected to persistent storage devices;
an internal network connecting the host systems to the storage
servers;

wherein execution of said software module further provides a
plurality of functions comprising:
translating all application I/O requests directed to the pool into
its physical device components;
forwarding the appropriate request to each individual storage device;
statistically timing and tracking I/O requests to the individual storage devices in order to compute storage device characteristics, including average I/O latencies and average sustained I/O throughput to the pool; and
querying the manager to obtain the collected and calculated configuration data for a selected storage pool.

12. The distributed storage system of Claim 10, wherein execution of the software module provides a plurality of functions comprising:
performed one of two types of data calculations from among: (1) translating information collected from each individual storage device into data applicable to the entire pool; and (2) generating data that is not obtainable from an individual storage device and data that is difficult to extrapolate to a pool basis;
providing an interface for access by applications and system administrators to the configuration data, such that (1) each application is able to query the interface to automatically match functions to the appropriate storage pool;
presenting the information to the system administrators for manually matching functions to storage pools; and
automatically assigning permissions to the interface to enable said interface to be accessible to all host system applications with said appropriate permissions.

13. A computer program product comprising:
a computer readable medium; and
program code on said computer readable medium for:
querying one or more storage devices connect to a host system for configuration information of the one or more storage devices; and
responsive to receipt of the configuration information, automatically assigning specific ones of the one or more storage devices to an application running on the host system, based on said configuration information.

14. The computer program product of Claim 13, further comprising program code for:
analyzing the configuration information received from the one or more storage devices to determine an optimal allocation of storage devices to the application; and
wherein said assigning further selects specific ones of the one or more storage devices that provide said optimal allocation.

15. The computer program product of Claim 13, wherein:
said distributed storage system comprises one or more host systems locally connected to persistent storage devices, one or more storage servers connected to persistent storage devices and an internal network connecting the host systems to the storage servers; and
said program code comprises code for:
grouping said one or more storage devices into pools;
maintaining, via a management utility of the host component, a knowledge base of (1) a physical device composition of each pool of storage devices and (2) interaction with software modules controlling the storage devices; and
collecting, calculating and distributing requisite configuration information of each storage device via a device configuration module of said management utility.

16. The computer program product Claim 13, further comprising code for:
collecting configuration data of the storage devices by automatically initiating the collection of obtainable configuration information when the storage device is opened; and
when a first device provides at least one of different formats and recording of configuration data in a particular class from a second device, normalizing said configuration data into a pre-established format for recording said configuration data; wherein, said configuration module is enhanced with device specific modules for the collection and translation of device specific data, and wherein said code for normalizing includes code for allocating a maximum transfer size for the pool as the minimum of the maximum transfer sizes over all the individual storage devices.

17. The method of Claim 13, further comprising code for performing one of two types of data calculations from among:
translating information collected from each individual storage device into data applicable to the entire pool; and
generating data that is not obtainable from an individual storage device and data that is difficult to extrapolate to a pool basis.

18. The computer program product of Claim 17, further comprising program code for:
translating all application I/O requests directed to the pool into its physical device components; and forwarding the appropriate request to each individual storage device.

19. The computer program product of Claim 13, further comprising program code for:

- statistically timing and tracking I/O requests to the individual storage devices in order to compute storage device characteristics, including average I/O latencies and average sustained I/O throughput to the pool; and
- querying the manager to obtain the collected and calculated configuration data for a selected storage pool.

20. The computer program product of Claim 19, further comprising program code for:

- providing an interface for access by applications and system administrators to the configuration data, such that (1) each application is able to query the interface to automatically match functions to the appropriate storage pool;
- presenting the information to the system administrators for manually matching functions to storage pools; and
- automatically assigning permissions to the interface to enable said interface to be accessible to all host system applications with said appropriate permissions.
Fig. 2
Fig. 3

1. DETECT STORAGE DEVICE CONNECTED TO HOST SYSTEM

2. GENERATE AND ISSUE QUERY TO STORAGE DEVICE

3. CONFIG, MFG FOR STORAGE DEVICE AVAILABLE?
   - NO: PERFORM 2ND DATA CALCULATION
   - YES: RETRIEVE CONFIG. INFO FROM STORAGE DEVICE

4. NORMALIZE CONFIG. INFO

5. PERFORM 1ST DATA CALCULATION

6. INITIATE DATA DISTRIBUTION
**INTERNATIONAL SEARCH REPORT**

**International application No**
PCT/EP2006/065023

**A CLASSIFICATION OF SUBJECT MATTER**
INV. G06F9/50
ADD. G06F3/06

**INTERNATIONAL SEARCH REPORT**

**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 practical search terms used)

**EPO-Internal**

**C DOCUMENTS_considered TO BE RELEVANT**

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document with indication where appropriate of the relevant passages</th>
<th>Relevant to claim No</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>CARLSON M A - &quot;Create automated and distributed management applications with Jiro technology&quot; JAVA WORLD, UEB PUB., SAN FRANCISCO, CA, US, 21 February 2000 (2000-02-21), pages 1-6, XP002273325 ISSN: 1091-8906 the whole document</td>
<td>1,9,10, 12,13,20</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C

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 document but published on or after the international filing date
  *L* document which may throw doubts on prior art 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
  *Z* document member of the same patent family

Date of the actual completion of the international search
18 December 2006

Date of mailing of the international search report
29/12/2006

Name and mailing address of the ISA:
European Patent Office
P B 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel (+31-70) 340-2040 Tx 31 651 epp sl
Fax (+31-70) 340-3016

Authorized officer
Michel, Thierry
# INTERNATIONAL SEARCH REPORT

## INTERNATIONAL APPLICATION

**Application No:**

PCT/EP2006/065023

**CONTINUATION (A):**

**DOCUMENTS CONSIDERED TO BE RELEVANT**

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>US 2003/014911 A2 (SUN MICROSYSTEMS INC [US]) 20 February 2003 (2003-02-20) paragraphs [0013], [0014], [0023], [0024], [0064]</td>
<td>1,9,10,12,13,20</td>
</tr>
<tr>
<td>A</td>
<td>EP 1 372 073 A2 (HEWLETT PACKARD CO [US]) 17 December 2003 (2003-12-17) paragraphs [0011], [0013], [0016]</td>
<td>4,6,10,16</td>
</tr>
</tbody>
</table>

Form PCT/ISA/210 (continuation of second sheet) (April 2005)
<table>
<thead>
<tr>
<th>Patent document cited in search report</th>
<th>Publication date</th>
<th>Patent family member(s)</th>
<th>Publication date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>WO 03025756 A2</td>
<td>27-03-2003</td>
</tr>
<tr>
<td>US 2003061129 A1</td>
<td>27-03-2003</td>
<td>NONE</td>
<td></td>
</tr>
<tr>
<td>GB 2351375 A</td>
<td>27-12-2000</td>
<td>CN 1268703 A</td>
<td>04-10-2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DE 10014448 A1</td>
<td>16-11-2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IE 20000203 A1</td>
<td>21-02-2001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>JP 2000339098 A</td>
<td>08-12-2000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003033398 A1</td>
<td>13-02-2003</td>
</tr>
<tr>
<td>EP 1372073 A2</td>
<td>17-12-2003</td>
<td>CN 1470987 A</td>
<td>28-01-2004</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2003236822 A1</td>
<td>25-12-2003</td>
</tr>
</tbody>
</table>