EXHIBIT 4

US008099420B2

(12) **United States Patent**
Farber et al.

(10) **Patent No.: US 8,099,420 B2**
(45) **Date of Patent: *Jan. 17, 2012**

(54) **ACCESSING DATA IN A DATA PROCESSING SYSTEM**

(75) Inventors: **David A. Farber**, Ojai, CA (US); **Ronald D. Lachman**, Northbrook, IL (US)

(73) Assignees: **PersonalWeb Technologies, LLC**, Tyler, TX (US); **Level 3 Communications, LLC**, Broomfield, CO (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 1312 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **11/017,650**

(22) Filed: **Dec. 22, 2004**

(65) **Prior Publication Data**

US 2005/0114296 A1 May 26, 2005

Related U.S. Application Data

(60) Continuation of application No. 09/987,723, filed on Nov. 15, 2001, now Pat. No. 6,928,442, which is a continuation of application No. 09/283,160, filed on Apr. 1, 1999, now Pat. No. 6,415,280, which is a division of application No. 08/960,079, filed on Oct. 24, 1997, now Pat. No. 5,978,791, which is a continuation of application No. 08/425,160, filed on Apr. 11, 1995, now abandoned.

(51) **Int. Cl.**
G06F 17/30 (2006.01)

(52) **U.S. Cl.** **707/758**; 707/781; 707/821

(58) **Field of Classification Search** 707/758, 707/781, 821

See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

3,668,647	A	6/1972	Evangelisti
3,835,260	A	9/1974	Prescher et al.
4,096,568	A	6/1978	Bennett et al.
4,215,402	A	7/1980	Mitchell
4,221,003	A	9/1980	Chang et al.
4,290,105	A	9/1981	Cichelli
4,376,299	A	3/1983	Rivest
4,405,829	A	9/1983	Rivest
4,412,285	A	10/1983	Neches

(Continued)

FOREIGN PATENT DOCUMENTS

EP 0 268 069 A2 5/1988

(Continued)

OTHER PUBLICATIONS

Affidavit of Timothy P. Walker in Support of CWIS' Opening Markman Brief Construing the Terms at Issue in U.S. Patent No. 6,415,280, dated Jul. 25, 2003, from Civil Action No. 02-11430 RWZ.

(Continued)

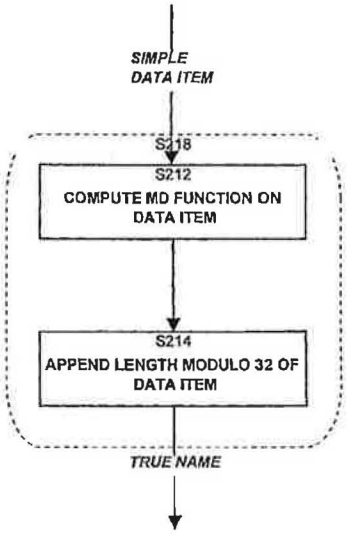
Primary Examiner — Khanh B Pham

(74) *Attorney, Agent, or Firm* — Davidson Berquist Jackson & Gowdey, LLP; Brian Siritzky

(57) **ABSTRACT**

Access to data items uses names based on the data in the data items; the name of a data item may be based, at least in part, on a function of some or all of the bits that comprise the data item. A data item may comprise an arbitrary sequence of bits. The function may include a hash function or a message digest function. The name of a data item may be compared to a list of names of other data items.

178 Claims, 31 Drawing Sheets



ACCESSING DATA IN A DATA PROCESSING SYSTEM

RELATED APPLICATIONS

This is a continuation of and claims priority to application Ser. no. 09/987,723, filed Nov. 15, 2001, now U.S. Pat. No. 6,928,442, issued Aug. 9, 2005 (the contents of which are hereby incorporated herein by reference), which is a continuation of application Ser. No. 09/283,160, filed Apr. 1. 1999, now U.S. Pat. No. 6,415,280, which is a division of application Ser. No. 08/960,079, filed Oct. 24, 1997, now U.S. Pat. No. 5,978,791 filed Oct. 24, 2001 which is a continuation of Ser. No. 08/425,160, filed Apr. 11, 1995, now abandoned.

BACKGROUND OF THE INVENTION

1. Field of the Invention

This invention relates to data processing systems and, more particularly, to data processing systems wherein data items are identified by substantially unique identifiers which depend on all of the data in the data items and only on the data in the data items.

2. Background of the Invention

Data processing (DP) systems, computers, networks of computers, or the like, typically offer users and programs various ways to identify the data in the systems.

Users typically identify data in the data processing system by giving the data some form of name. For example, a typical operating system (OS) on a computer provides a file system in which data items are named by alphanumeric identifiers. Programs typically identify data in the data processing system using a location or address. For example, a program may identify a record in a file or database by using a record number which serves to locate that record.

In all but the most primitive operating systems, users and programs are able to create and use collections of named data items, these collections themselves being named by identifiers. These named collections can then, themselves, be made part of other named collections. For example, an OS may provide mechanisms to group files (data items) into directories (collections). These directories can then, themselves be made part of other directories. A data item may thus be identified relative to these nested directories using a sequence of names, or a so-called pathname, which defines a path through the directories to a particular data item (file or directory).

As another example, a database management system may group data records (data items) into tables and then group these tables into database files (collections). The complete address of any data record can then be specified using the database file name, the table name, and the record number of that data record.

Other examples of identifying data items include: identifying files in a network file system, identifying objects in an object-oriented database, identifying images in an image database, and identifying articles in a text database.

In general, the terms "data" and "data item" as used herein refer to sequences of bits. Thus a data item may be the contents of a file, a portion of a file, a page in memory, an object in an object-oriented program, a digital message, a digital scanned image, a part of a video or audio signal, or any other entity which can be represented by a sequence of bits. The term "data processing" herein refers to the processing of data items, and is sometimes dependent on the type of data item being processed. For example, a data processor for a digital image may differ from a data processor for an audio signal.

In all of the prior data processing systems the names or identifiers provided to identify data items (the data items being files, directories, records in the database, objects in object-oriented programming, locations in memory or on a physical device, or the like) are always defined relative to a specific context. For instance, the file identified by a particular file name can only be determined when the directory containing the file (the context) is known. The file identified by a pathname can be determined only when the file system (context) is known. Similarly, the addresses in a process address space, the keys in a database table, or domain names on a global computer network such as the Internet are meaningful only because they are specified relative to a context.

In prior art systems for identifying data items there is no direct relationship between the data names and the data item. The same data name in two different contexts may refer to different data items, and two different data names in the same context may refer to the same data item.

In addition, because there is no correlation between a data name and the data it refers to, there is no a priori way to confirm that a given data item is in fact the one named by a data name. For instance, in a DP system, if one processor requests that another processor deliver a data item with a given data name, the requesting processor cannot, in general, verify that the data delivered is the correct data (given only the name). Therefore it may require further processing, typically on the part of the requester, to verify that the data item it has obtained is, in fact, the item it requested.

A common operation in a DP system is adding a new data item to the system. When a new data item is added to the system, a name can be assigned to it only by updating the context in which names are defined. Thus such systems require a centralized mechanism for the management of names. Such a mechanism is required even in a multi-processing system when data items are created and identified at separate processors in distinct locations, and in which there is no other need for communication when data items are added.

In many data processing systems or environments, data items are transferred between different locations in the system. These locations may be processors in the data processing system, storage devices, memory, or the like. For example, one processor may obtain a data item from another processor or from an external storage device, such as a floppy disk, and may incorporate that data item into its system (using the name provided with that data item).

However, when a processor (or some location) obtains a data item from another location in the DP system, it is possible that this obtained data item is already present in the system (either at the location of the processor or at some other location accessible by the processor) and therefore a duplicate of the data item is created. This situation is common in a network data processing environment where proprietary software products are installed from floppy disks onto several processors sharing a common file server. In these systems, it is often the case that the same product will be installed on several systems, so that several copies of each file will reside on the common file server.

In some data processing systems in which several processors are connected in a network, one system is designated as a cache server to maintain master copies of data items, and other systems are designated as cache clients to copy local copies of the master data items into a local cache on an as-needed basis. Before using a cached item, a cache client must either reload the cached item, be informed of changes to the cached item, or confirm that the master item corresponding to the cached item has not changed. In other words, a cache client must synchronize its data items with those on the

on the medium, but also records directory entries for each file in a frozen directory structure. By copying and modifying this directory, it is possible to create an on line patch, or small modification of an existing read-only file. For example, it is possible to create an online representation of a modified CD-ROM, such that the unmodified files are actually on the CD-ROM, and only the modified files are online.

In operation, the system tracks possession of specific data items according to content by owner, independent of the name, date, or other properties of the data item, and tracks the uses of specific data items and files by content for accounting purposes. Using the Track for Accounting Purposes extended mechanism provides a way to know reliably which files have been stored on a system or transmitted from one system to another.

True Names in Relational and Object-Oriented Databases

Although the preferred embodiment of this invention has been presented in the context of a file system, the invention of True Names would be equally valuable in a relational or object-oriented database. A relational or object-oriented database system using True Names would have similar benefits to those of the file system employing the invention. For instance, such a database would permit efficient elimination of duplicate records, support a cache for records, simplify the process of maintaining cache consistency, provide location-independent access to records, maintain archives and histories of records, and synchronize with distant or disconnected systems or databases.

The mechanisms described above can be easily modified to serve in such a database environment. The True Name registry would be used as a repository of database records. All references to records would be via the True Name of the record. (The Local Directory Extensions table is an example of a primary index that uses the True Name as the unique identifier of the desired records.)

In such a database, the operations of inserting, updating, and deleting records would be implemented by first assimilating records into the registry, and then updating a primary key index to map the key of the record to its contents by using the True Name as a pointer to the contents.

The mechanisms described in the preferred embodiment, or similar mechanisms, would be employed in such a system. These mechanisms could include, for example, the mechanisms for calculating true names, assimilating, locating, realizing, deleting, copying, and moving True Files for mirroring True Files, for maintaining a cache of True Files, for grooming True Files, and other mechanisms based on the use of substantially unique identifiers.

While the invention has been described in connection with what is presently considered to be the most practical and preferred embodiments, it is to be understood that the invention is not to be limited to the disclosed embodiment, but on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims.

What is claimed:

1. A computer-implemented method implemented at least in part by hardware in combination with software, the method comprising the steps of:

(A) obtaining a plurality of identifiers, each of said identifiers in said plurality of identifiers corresponding to at least one of a plurality of data items, each of said identifiers in said plurality of identifiers being based, at least in part, on a first given function of at least some of the data that comprise the contents of a corresponding one of the plurality of data items, wherein two identical data items of said plurality of data items have identical identifiers;

(B) responsive to a request, the request including at least a specific name for a particular sequence of bits, the specific name having been determined, at least in part, using a second given function of the particular sequence of bits, wherein two identical sequences of bits have the same name as determined using the second given function, and wherein the first given function is the same as the second given function that was used to determine the specific name for the particular sequence of bits, hardware in combination with software, ascertaining whether or not the specific name for the particular sequence of bits corresponds to an identifier in said plurality of identifiers; and,

(C) based at least in part on said ascertaining in step (B), selectively permitting at least one copy of the particular sequence of bits to be distributed across or accessed by or from a plurality of computers in a network, wherein a copy of the particular sequence of bits is not permitted to be distributed or accessed without authorization, as determined based, at least in part, on whether or not the specific name for the particular sequence of bits corresponds to an identifier in said plurality of identifiers.

2. A method as in claim **1**, wherein a copy of the particular sequence of bits is not permitted to be distributed on behalf of unlicensed parties or unauthorized parties.

3. A method as in claim **1**, wherein distribution of or accessing an unauthorized copy of a sequence of bits is not allowed.

4. A method as recited in claim **3** further comprising:

permitting at least one copy of the particular sequence of bits to be distributed in a network when it is not determined that the particular sequence of bits is unauthorized.

5. The method of claim **1** wherein distribution of or accessing an unlicensed copy of a sequence of bits is not allowed.

6. The method of claim **1** wherein the specific name for the particular sequence of bits corresponds to an identifier of the plurality of identifiers when the specific name for the particular sequence of bits exactly matches the identifier of the plurality of identifiers.

7. A method as recited in claim **1** wherein the specific name of the particular sequence of bits is based, at least in part, on a function of all of the bits of the particular sequence of bits.

8. The method of claim **1** further comprising:

denying permission for the particular sequence of bits to be distributed in a network when it is determined that access to the particular sequence of bits is not authorized.

9. The method of claim **1** further comprising:

maintaining the list plurality of identifiers.

10. The method of claim **9** wherein said maintaining the plurality of identifiers comprises:

adding new identifiers to said plurality of identifiers.

11. The method of claim **1** further comprising:

using the specific name for the particular sequence of bits to determine whether the particular sequence of bits is present in a data processing system.

12. The method of claim **1** wherein the particular sequence of bits represent data selected from the group comprising: a file, a portion of a file, a page in memory, a digital message, a portion of a digital message, a digital image, a portion of a digital image, a video signal, a portion of a video signal, an audio signal, a portion of an audio signal, a software product, a portion of a software product, and a identifier of a data item.

13. A computer-implemented method, implemented at least in part by hardware in combination with software, the method comprising the steps:

(A) obtaining a specific name for a particular sequence of bits, the specific name having been determined at least in part as a first given function of at least some of the sequence of bits, wherein two identical sequences of bits will have the same name, as determined using the first given function; and

(B) ascertaining, by hardware in combination with software, whether or not the specific name for the particular sequence of bits corresponds to an identifier in a plurality of identifiers, said plurality of identifiers corresponding to a plurality of data items, each of said plurality of identifiers being based, at least in part, on a second given function of the contents of a corresponding one of the plurality of data items, wherein two identical data items have identical identifiers, as determined by said second given function, and wherein the second given function is the same as the first given function; and

(C) based at least in part on said ascertaining in step (B), selectively allowing a copy of the particular sequence of bits to be distributed to or provided or accessed by or from at least one of the computers in said plurality of computers, wherein a copy of the sequence of bits is not to be distributed or provided or accessed without authorization, as determined based, at least in part, on whether or not the specific name for the particular sequence of bits corresponds to one of said plurality of identifiers.

14. A method as in claim 13, further comprising:

maintaining accounting information relating to at least some data items in the system; and

using the accounting information as a basis for charges based on an identity of the at least some data items.

15. A method as in claim 13, further comprising:

collecting information relating to the particular sequence of bits.

16. The method of claim 13 wherein the specific name for the particular sequence of bits is based, at least in part, on a size or length of the particular sequence of bits.

17. The method of claim 13 wherein the specific name for the particular sequence of bits comprises a digital fingerprint of the particular sequence of bits.

18. The method of claim 13 wherein the specific name for the particular sequence of bits is a True Name of the particular sequence of bits.

19. The method of any one of claims 1 and 13 wherein the specific name for the particular sequence of bits is a based on all of the bits and only the bits in the particular sequence of bits.

20. The method of claim 13 further comprising:

obtaining a copy of the particular sequence of bits; and

wherein the step (A) of obtaining the specific name for the particular sequence of bits comprises:

hardware in combination with software, determining the specific name for the particular sequence of bits using the copy of the particular sequence of bits.

21. The method of claim 13 further comprising:

obtaining the plurality of identifiers.

22. The method of claim 13 wherein, when contents of the particular sequence of bits represent an audio signal or a portion of an audio signal, the specific name of the particular sequence of bits is a function of at least some of the data comprising the audio signal; and

when contents of the particular sequence of bits represent a video signal or a portion of a video signal, the specific name of the particular sequence of bits is a function of at least some of the data comprising the video signal.

23. The method of any one of claims 1 and 13, wherein the specific name for the particular sequence of bits comprises a request for the particular sequence of bits.

24. The method of any one of claims 1 and 13, wherein the specific name for the particular sequence of bits was determined using only bits in the sequence of bits.

25. A computer-implemented method implemented at least in part by hardware in combination with software, the method comprising the steps:

(A) hardware in combination with software, determining a first content-dependent name for a particular sequence of bits, at least in part by applying a particular function to at least some of the particular sequence of bits, said particular function comprising a message digest function or a hash function, wherein two identical sequences of bits will have the same content-dependent name as determined using said particular function;

(B) ascertaining whether or not said first content-dependent name for the particular sequence of bits corresponds to one of a plurality of identifiers, said plurality of identifiers corresponding to a plurality of data items, each identifier of said plurality of identifiers being based, at least in part, on a first given function of the data that comprise the contents of a corresponding one of the plurality of data items, wherein said first given function comprises the particular function used to determine the first content-dependent name for said particular sequence of bits: and,

(C) based at least in part on said ascertaining in step (B), selectively allowing a copy of the particular sequence of bits to be provided to or accessed by or from at least one of the computers in a network of computers, wherein a copy of the sequence of bits is not to be provided or accessed without authorization, as determined, at least in part, based on whether or not said first content-dependent name of the particular sequence of bits corresponds to one of the plurality of identifiers.

26. The method as in 25 wherein the plurality of identifiers is in a table comprising said plurality of identifiers.

27. The method as in 25 wherein the plurality of identifiers is in a database comprising said plurality of identifiers.

28. The method of claim 25 wherein the first content-dependent name for the particular sequence of bits is based, at least in part, on a size or length of the particular sequence of bits.

29. The method of claim 25 wherein the first content-dependent name for the particular sequence of bits comprises a digital fingerprint of the particular sequence of bits.

30. The method of claim 25 wherein the first content-dependent name for the particular sequence of bits is a True Name of the particular sequence of bits.

31. The method of claim 22 wherein the first content-dependent name for the particular sequence of bits is a based on all of the bits and only the bits in the particular sequence of bits.

32. The method of claim 25 where said first content-dependent name of the particular sequence of bits corresponds to an identifier of the first plurality of identifiers when said first content-dependent name of the particular sequence of bits exactly matches the identifier of the first plurality of identifiers.

33. The method of claim 25 further comprising:

(B2) obtaining a second content-dependent name for said at least one particular sequence of bits, the second content-dependent name being based at least in part on a

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.