(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
1 May 2003 (01.05.2003)

PCT

(10) International Publication Number
WO 03/036521 A1

(51) International Patent Classification⁷: G06F 17/30, 12/00, 15/80, 15/16, H04M 3/42, H04L 12/28

(21) International Application Number: PCT/US02/34309

(22) International Filing Date: 24 October 2002 (24.10.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/386,487 24 October 2001 (24.10.2001) US

(71) Applicant: **BEA SYSTEMS, INC.** [US/US]; 2315 North First Street, San Jose, CA 95131 (US).

(72) Inventors: **SELMAN, Daniel**; Kingsley House, Hollybush Lane, Stodmarsh, Kent, CT3 4BG (US). **BERGMAN, Robert**; 9669 Orangewood Drive, Denver, CO 80260 (US). **O'NEIL, Edward, K.**; 3455 Tabel Mesa Drive-Apartment G162, Boulder, CO 80305 (US).

(74) Agents: **MEYER, Sheldon, R.** et al.; Fliesler Dubb Meyer & Lovejoy LLP, Four Embarcadero Center, Suite 400, San Francisco, CA 94111-4156 (US).

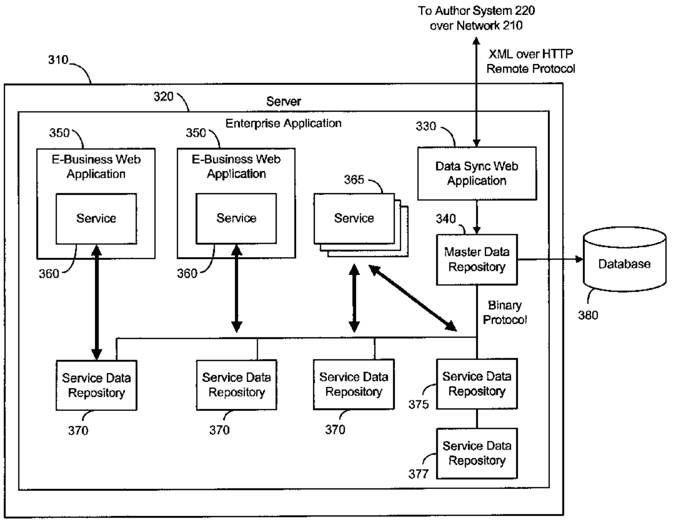
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

[Continued on next page]

(54) Title: DATA SYNCHRONIZATION



(57) **Abstract:** Application data used by enterprise applications (320) running on one or more servers (230,240,250) can be synchronized with application data deployed from other locations. Application data can be transmitted as XML files over HTTP to facilitate synchronization with web-enabled servers. Multiple service data repositories (370,375) can be synchronized with application data residing in a master data repository (340) that runs within an enterprise application. Server-to-server data synchronization functionality is also provided through the use of proxy data repositories. Application data files can be deployed from an author system (220) to remote servers. Newly-authored application data files can be versioned in the author system (220) using a source control system and selectively deployed to various servers by the author in conjunction with a multi-stage testing process in preparation for deployment of the application data to a production server.

TITLE

Data Synchronization

COPYRIGHT NOTICE

CROSS-REFERENCE TO RELATED APPLICATIONS

The present application claims the benefit of the filing date of U.S. Provisional Patent Application No. 60/386,487 (Attorney Docket No. BEAS-1112US0), filed October 24, 2001, incorporated by reference herein.

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to technology for deploying and synchronizing data.

2. Description of the Related Art

Web-based software applications have emerged as powerful tools which provide valuable services to vendors and customers alike. E-business web applications can be configured to operate in accordance with business logic to implement a variety of processes which facilitate electronic commerce. Such applications may also provide services which interact with remote Internet clients and/or applications.

The JAVA™ 2 Platform Enterprise Edition (J2EE), available from Sun Microsystems, Inc. of Santa Clara, California, facilitates the development of electronic business web applications that run within the context of multi-tier enterprise applications. The operation of such web applications is governed by the data ("application data") used to configure the web applications. As a result, the functionality provided by web applications can be modified by updating the application data associated with the applications.

Unfortunately, the deployment and synchronization of new or updated application data to J2EE web applications can be a cumbersome process. Newly-authored application data is often stored in a database in accordance with a particular schema used by the database. If a developer seeks to deploy such application data to a running production server, the data must be extracted from the database before being converted into a form suitable for transmission to the server. Scripts are often necessary to export the application data from a database into a file system. Scripts may also be necessary to import the application data to a running server. For these reasons, maintaining different versions of application data can also be an awkward and unwieldy endeavor.

Moreover, before new application data can be moved to a running production server, it may be necessary to deploy the application data to multiple servers to comply with a multi-stage testing process. This transfer of application data can be a cumbersome manual process which becomes increasingly error-prone as application data is transferred from server to server and more persons are involved. Thus, prior techniques of deploying and synchronizing application data in the context of the J2EE platform can be largely ad-hoc and undesirable.

SUMMARY OF THE INVENTION

The present invention, roughly described, is directed to synchronizing application data used by enterprise applications running on one or more servers. For example, in one embodiment, application data is received and synchronized with a database and data repository. In another embodiment, an application data deployment method is provided allowing application data to be authored, submitted to a source control system, and sent to a remote server where a data repository of the server is synchronized with the data.

In another embodiment, application data is synchronized between enterprise applications through the use of proxy data repositories. In another embodiment, application data is synchronized between data repositories through the polling of one data repository by another.

Systems and computer readable media are also provided for implementing portions of, and variations of, these methods. Many other embodiments are also possible, as set forth in the present disclosure.

BRIEF DESCRIPTIONS OF THE DRAWINGS

Figure 1 illustrates a conceptual diagram of a system for creating and versioning application data files in accordance with an embodiment of the present invention.

Figure 2 illustrates a high level block diagram of a system for synchronizing data in accordance with an embodiment of the present invention.

Figure 3 illustrates a block diagram of a server running an enterprise application having data that can be synchronized in accordance with an embodiment of the present invention.

Figure 4 is a flowchart illustrating a process for synchronizing data in accordance with an embodiment of the present invention.

Figure 5 illustrates a block diagram of multiple servers running enterprise applications with data that can be synchronized using a proxy data repository in accordance with an embodiment of the present invention.

Figure 6 illustrates a block diagram of a cluster of servers running enterprise applications with data that can be synchronized by an administration server in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

Figure 1 illustrates a conceptual diagram of a system for creating and versioning application data files in accordance with an embodiment of the present invention. Author 110 is a person with the responsibility of creating application data to be used by one or more web applications and/or services. In various embodiments, author 110 can be a line-of-business manager, business engineer, developer, or other person responsible for authoring application data. The application data can be used to configure a web application to operate in accordance with business logic identified by author 110. For example, application data created by author 110 can include business policy documents such as campaign definitions, scenarios, business rule sets, portal definitions, and/or other types of data used by web applications.

In order to produce the application data, author 110 can interface with various software tools, such as control center 120. In one embodiment, control center 120 is an E-Business Control Center tool available from BEA Systems, Inc. of San Jose, California. The data created by author 110 using control center 120 can be stored in a local hierarchical file system as a set of application data files. Figure 1 illustrates an example of application data files 130 stored in a hierarchical file system. The application data files 130 can be organized according to the web application they are created for. In one embodiment, the application data created with control center 120 is stored in XML files in accordance with an XML schema which can be interpreted by the web application for which the data is authored. Since XML can be read and interpreted by various systems across multiple platforms, the use of XML application data provides advantages over database scripting-oriented approaches.

Application data files 130 produced by the interaction of author 110 with control center 120 can be checked into source control system 140. Source control system 140 allows different versions of the application data files 130 to be easily recalled for testing, deployment, synchronization, and/or other purposes. For example, if faulty application data is deployed to a web application, the source control system 140 allows author 110 to recall a previous version of the application data and re-deploy the data to the web application. The use of source control system 140 can also prevent multiple authors from overwriting each other's files when application data is being authored concurrently. In one embodiment, source control system 140 is any suitable source control system known in the art.

After application data has been created, it will typically be subject to a multi-stage approval process to test the interaction of the newly-authored application data with other data and/or applications created by other authors. For example, before application data is deployed to a production server, it may be subject to separate development and quality assurance stages. These stages help ensure that the web applications running on live production servers will operate reliably.

During this collaborative development of web applications, it can become desirable for software developers to synchronize application data across enterprise application boundaries to remote servers. Various data synchronization processes further described herein leverage the advantages provided by HTTP to facilitate the synchronization of application data on remote servers accessible via the Internet. In accordance with certain embodiments of the present invention, newly-created application data can be synchronized with one or more remote running servers. By identifying the URL associated with a server, application data can be remotely synchronized to any server accessible over the Internet via HTTP.

Figure 2 illustrates a high level block diagram of a system for synchronizing data in accordance with an embodiment of the present invention. The system of Figure 2 can be used for collaboratively developing and testing web applications before they are ultimately deployed on a production server. In one embodiment, the system of Figure 2 is implemented using one or more components of BEA WebLogic Platform^{TM}, available from BEA Systems, Inc. of San Jose, California. As illustrated in Figure 2, a plurality of servers are in communication with network 210. In one embodiment, the system of Figure 2 is implemented as a peer-to-peer network. Each server of Figure 2 is associated with a database which can store application data used by web applications and/or services running on the server. As indicated by Figure 2, each server can be used for a different purpose in a multi-stage application data approval process. For example, these purposes can include: development (server 230 and database 235), quality assurance (server 240 and database 245), production (server 250 and database 255), and others (not shown).

Each server of Figure 2 is capable of receiving application data from author system 220 over network 210. The application data received from author system 220 can be synchronized with application data used by web applications running on the various servers as further described herein. In one embodiment, author system 220 is the system illustrated in Figure 1.

Server tools 260 and database tools 265 are also illustrated in Figure 2. These tools allow for the realtime modification of data on production server 250 and production database 255, respectively. In one embodiment, server tools 260 are JSP-based tools which allow changes to be made directly to production server 250.

Figure 3 illustrates a block diagram of a server running an enterprise application having data that can be synchronized in accordance with an embodiment of the present invention. Server 310 can be any of the servers illustrated in Figure 2 capable of receiving application data from author system 220. As illustrated in Figure 3, a plurality of web applications and services reside in enterprise application 320 running on server 310. E-business web applications 350 include services 360 that rely upon application data stored in service data repositories 370, 375, and/or 377. Additional services 365 running within the context of enterprise application 320 can also rely upon application data stored in the service data repositories. In one embodiment, the service data repositories are Java objects that function as in-memory caches of data items of interest to a deployed class of service.

Data sync web application 330 also runs within enterprise application 320 and is responsible for performing data synchronization operations in accordance with the present invention. As indicated by Figure 3, data sync application 330 can communicate with author system 220 of Figure 2 over network 210. A master data repository 340 within enterprise application 320 can communicate with data sync application 330, database 380, and data repositories 370 and 375.

Data sync application 330 can intercept incoming synchronization requests sent from author system 220 over network 210. In one embodiment, these synchronization requests are sent as XML over HTTP in accordance with a remote protocol. When data sync application 330 receives a synchronization request from author system 220, it channels application data received from author system 220 to master data repository 340 which is a runtime representation of the application data. The data sync application 330 also channels the application data received to database 380 which is a persistent storage location for the application data. In one embodiment, data repository 340 stores incoming application data as Enterprise Java Beans (EJBs). In another embodiment, database 380 is a relational database management system.

In one embodiment, data sync application 330 includes a set of JSPs that are capable of reporting: the contents of master data repository 340, data repositories that are successfully synchronized with master data repository 340, and the contents of each data repository in enterprise application 320.

The service data repositories of Figure 3 each contain subsets of the application data stored in master data repository 340. Services 360 and 365 use the data in the service data repositories to carry out business tasks. For example, a rules service can use the data to evaluate whether customers fit into a particular customer segment. Since the service data repositories of Figure 3 maintain only a subset of the data in master data repository 340, each service is able to parse a small, specific set of application data needed by the service without having to maintain copies of all application data maintained in master data repository 340.

Each time the master data repository 340 is synchronized, it notifies data repositories 370 and 375. These data repositories then synchronize their data subsets with the data in the master. In one embodiment, application data is communicated between the components of enterprise application 320 in accordance with a binary protocol. Multiple enterprise applications (not shown) running on server 310 can also be synchronized using a separate data sync application running in each enterprise application. Proxy data repositories can be used to facilitate such synchronizations.

In some cases, the master data repository 340 does not directly synchronize data with a data repository. Rather, a notification chain can be employed using multiple data repositories. Referring to Figure 3, data repositories 375 and 377 form a notification chain. When data repository 375 receives updated application data from master data repository 340, it notifies data repository 377. Data repository 377 can then be synchronized with the application data received by data repository 375. As a result, both repositories 375 and 377 can be synchronized without master data repository 340 having to send updated application data to both repositories.

Figure 4 is a flowchart illustrating a process for synchronizing data in accordance with an embodiment of the present invention. When application data is to be synchronized from author system 220 to any of the servers of Figure 2, a synchronization process can be initiated (step 410) by sending a synchronization request from author system 220 to the URL of the server to be synchronized where it will be intercepted by data sync application 330. In various embodiments, author 110 can initiate the synchronization process using a graphical interface of control center 120 or a Java command line.

In various embodiments of the present invention, different synchronization modes can be employed. In a refresh-from-client mode, all application data for a given application is synchronized. Using this mode, all data is cleared from the data repositories and then all relevant application data files are sent from the author's local file system to a data sync web application. This mode can be helpful to synchronize in-memory data with persisted data, or to recover from a synchronization error. In a one-way-from-client mode, only those application data files that have been removed, updated, or created on the author's local file are updated. Thus, the amount of data transmitted between an author system and a server to be synchronized can be minimized.

Upon initiation of the synchronization process in step 410, application data to be synchronized will be sent from author system 220 to server 310 over network 210 (step 420). In one embodiment, the application data is sent as XML files over HTTP using a POST command. The application data is received by data sync application 330. Data sync application 330 then synchronizes database 380 and master data repository 340 with the newly-received application data (step 430). As a result of step 430, database 380 contains the updated application data stored in a database format for persistent storage, and master data repository 340 contains the updated application data stored in an EJB runtime format.

As previously described herein, the various applications and services of Figure 3 may rely upon application data resident in service data repositories. Accordingly, each of the service data repositories can subscribe to receive updated application data. In order to update the application data in data repositories 370 and 375, master data repository 340 notifies each of service data repositories 370 and 375 of any updated application data for which the repositories have subscribed to receive (step 440). Service data repository 377 receives its notification via a notification chain, as previously described. Master data repository 340 also notifies proxy data repositories (not shown) for enterprise applications running on other servers, as further described herein. After the service data repositories have been notified of the updated application data, each repository synchronizes itself with the particular application data in master data repository 340 to which it subscribes (step 450). Proxy data repositories also forward any notifications sent in step 440 to their remote data sync web applications in order to synchronize repositories running in other enterprise applications (not shown) on server 310 or other remote servers (step 460), as further described herein.

Service data repositories 370 and/or 375 can also poll master data repository 340 periodically to check whether application data has been updated. If updated data is detected, then application data maintained in master data repository 340 can be synchronized with service data repositories 370 and/or 375.

Master data repository 340 maintains a log that describes each update and includes a description of which data repositories were successfully updated, and which were not. Data sync application 330 can return a status message to author system 220 via HTTP or a Java command shell to identify which application data files were successfully synchronized. Each data repository of Figure 3 can also maintain logs that describe its current contents.

As illustrated in Figure 3, application data can be synchronized between an author system and an enterprise application running on a single server. In accordance with various embodiments of the present invention, multiple servers can also synchronized through a single synchronization process initiated in step 410 of Figure 4. This synchronization of multiple servers can be facilitated through the use of additional elements as set forth in Figure 5.

Figure 5 illustrates a block diagram of multiple servers running enterprise applications with data that can be synchronized using a proxy data repository in accordance with an embodiment of the present invention. Enterprise applications 620 and 660 are running on servers 610 and 650, respectively. A master data repository 630 running within enterprise application 620 can be synchronized by a data sync application (not shown) in communication with an author system (not shown) as previously described herein. Master data repository 630 is in communication with proxy data repository 640 also running within enterprise application 620. Proxy data repository 640 can subscribe to receive updated application data stored in master data repository 630.

Data sync web application 680 is an instance of a data sync application that runs on remote server 650. However, rather than receiving updates directly from an author system 220 (similar to data sync application 330 of Figure 3), data sync application 680 receives updates sent from proxy data repository 640. Remote master data repository 670 is a master data repository for enterprise application 660 which is updated in accordance with application data received by data sync application 680. Remote master data repository 670 can pass updated application data to other proxy and/or service data repositories (not shown) running in enterprise application 660 as previously discussed herein. It will be understood that additional elements (not shown) can be present on the servers of Figure 5, such as one or more of the elements set forth in the servers of Figure 3 and/or Figure 6.

In operation, master data repository 630 receives updated application data from a data sync web application in communication with an author system (not shown) such as author system 220 of Figure 2. Proxy data repository 640 is subscribed to receive updated application data from master data repository 640. As a result, the updated application data received by master data repository 630 is passed to proxy data repository 640. Proxy data repository 640 then passes the application data from server 610 to the data sync application running on server 650. In one embodiment, application data is received by proxy data repository 640 in accordance with a binary protocol. Accordingly, proxy data repository 640 marshals the application data into an XML format before passing the application data as XML over HTTP to data sync application 680 in remote server 650.

When the updated application data is received by data sync application 680, the data sync application 650 proceeds to update remote master data repository 670 with the updated application data. A persistent storage database (not shown in Figure 5) as illustrated in Figure 3 is also synchronized, and the updated application data is then forwarded to all data repositories (not shown) of enterprise application 660 which are subscribed to receive the updated application data.

Thus, by subscribing proxy data repositories to receive updates from a master data repository, application data within enterprise applications running on remote servers can be synchronized with application data received by the master data repository. Proxies can also be used to synchronize data across different enterprise applications running on the same server. Moreover, by linking remote servers to additional remote servers through proxies, chains of multiple servers can be synchronized. These principles can be further applied to the synchronizing application data across clusters of servers as set forth in Figure 6.

Figure 6 illustrates a block diagram of a cluster of servers running enterprise applications with data that can be synchronized by an administration server in accordance with an embodiment of the present invention. Data sync web application 730 and master data repository 740 of enterprise application 720 on administration server 710 can communicate with managed servers 760 of cluster 790. As previously discussed, application data residing in data repositories (not shown) of enterprise applications 770 running on remote servers 760 can be synchronized with updated application data stored in master data repository 740. Master data repository 740 can pass updated application data to a proxy data repository (not shown) which then passes the data to data sync web applications (not shown) running on enterprise applications 770. As a result, multiple servers 760 can be synchronized by a single administration server 710 that receives updated application data from an author system (not shown). It will be understood that additional elements (not shown) can be present on the servers of Figure 6, such as one or more of the elements set forth in the servers of Figure 3 and/or Figure 5.

Where applicable, the present invention can be implemented using hardware, software, or combinations of hardware and software. Software in accordance with the present invention, such as program code and/or data, can stored on one or more computer readable mediums. Also where applicable, the various hardware components and/or software components set forth herein can be combined into composite components comprising software, hardware, or both without departing from the spirit of the present invention. Similarly, where applicable, the various hardware components and/or software components set forth herein can be dissected into sub-components comprising software, hardware, or both without departing from the spirit of the present invention. In addition, where applicable, it is contemplated that software components can be implemented as hardware components, and vice-versa. Furthermore, where applicable, the various steps set forth herein can be combined into composite steps and/or dissected into sub-steps. It is also contemplated that software components set forth herein can be implemented using one or more general purpose or specific purpose computers and/or computer systems, networked and/or otherwise.

The foregoing disclosure is not intended to limit the present invention to the precise forms or particular fields of use disclosed. It is contemplated that various alternate embodiments and/or modifications to the present invention are possible in light of the disclosure.

CLAIMS

What is claimed is:

1. A method for synchronizing application data on a server, comprising:

receiving application data to be synchronized;

synchronizing a database with said application data; and

synchronizing a first data repository with said application data.

2. The method of claim 1, wherein:

said first data repository is a service data repository.

3. The method of claim 1, further comprising:

notifying a second data repository of at least a subset of said application data; and

synchronizing said second data repository with said subset of said application data.

4. The method of claim 3, wherein:

said second data repository is a proxy data repository.

5. The method of claim 3, wherein:

said second data repository is subscribed to receive said subset of said application data from said first data repository.

6. The method of claim 1, wherein:

said first data repository comprises a runtime representation of said application data.

7. The method of claim 1, wherein:

said database provides persistent storage of said application data.

8. The method of claim 1, wherein:

said synchronizing steps are performed by a J2EE enterprise application.

9. The method of claim 1, wherein:

said application data is used by a service running in an enterprise application.

10. The method of claim 9, wherein:

said service is used by a web application.

11. The method of claim 1, wherein:

said application data defines business logic of a web application.

12. The method of claim 1, wherein:

said application data comprises at least one XML file.

13. The method of claim 1, wherein:

said application data comprises application data files adapted for transmission over HTTP.

14. The method of claim 1, wherein:

said method is performed as part of a testing process in preparation for deployment of said application data to a production server.

15. A method for deploying application data to a remote web server over a network, comprising:

authoring application data;

submitting said application data to a source control system;

identifying a server to receive application data; and

sending application data to said server, said server capable of synchronizing a data repository of said server with said application data.

16. The method of claim 15, wherein:

said server is identified by a URL associated with said server.

17. The method of claim 15, wherein:

said application data comprises at least one XML file.

18. The method of claim 15, wherein:

said application data comprises application data files adapted for transmission over HTTP.

19. The method of claim 15, wherein:

said method is performed as part of a testing process in preparation for deployment of said application data to a production server.

20. A method for synchronizing application data between enterprise applications, comprising:

receiving application data;

notifying a proxy data repository of said application data, said proxy data repository running in a first enterprise application;

sending said application data to said proxy data repository; and

forwarding said application data from said proxy data repository to a second enterprise application.

21. The method of claim 20, wherein:

said first and second enterprise applications reside in a single server.

22. The method of claim 20, wherein:

said first and second enterprise applications reside in separate servers.

23. The method of claim 20, wherein:

said application data comprises at least one XML file.

24. The method of claim 20, wherein:

said application data comprises application data files adapted for transmission over HTTP.

25. The method of claim 20, wherein:

said method is performed as part of a testing process in preparation for deployment of said application data to a production server.

26. A method for synchronizing data in a data repository of an enterprise application, comprising:

polling a master data repository;

determining whether application data stored in a subscribed data depository is synchronized with said master data depository; and

synchronizing said subscribed data depository with said master data depository in response to said determining step.

27. The method of claim 26, wherein:

said application data is in an EJB runtime format.

28. The method of claim 26, wherein:

said method is performed as part of a testing process in preparation for deployment of said application data to a production server.

29. A system for synchronizing application data, comprising:

a web server;

a data synchronization application running on said web server,

said data synchronization application capable of receiving application data sent over an Internet connection; and

a data repository in communication with said data synchronization application,

said data repository adapted for maintaining a runtime representation of said application data,

said data synchronization application capable of synchronizing said received application data with said data repository.

30. The system of claim 29, wherein:

said data synchronization application runs within an enterprise application of said server.

31. The system of claim 30, wherein:

said enterprise application is a J2EE enterprise application.

32. The system of claim 29, wherein:
said server is a development server.

33. The system of claim 29, wherein:
said server is a testing server.

34. The system of claim 29, wherein:
said server is a production server.

35. The system of claim 29, wherein:
said application data comprises at least one XML file.

36. The method of claim 29, wherein:
said application data comprises application data files adapted for transmission over HTTP.

37. A system for testing application data, comprising:
a plurality of web servers;
an author system in communication with said servers over the Internet;
a data synchronization application running on at least one of said servers, said data synchronization application capable of synchronizing application data received from said author system with application data of said server.

38. The system of claim 37, wherein:
said application data comprises at least one XML file.

39. The method of claim 37, wherein:
said application data comprises application data files adapted for transmission over HTTP.

40. A computer readable medium, comprising:
a storage medium; and

computer readable code embodied on said storage medium, said computer readable code for programming a computer to perform a method for synchronizing application data on a server, the method comprising:

receiving application data to be synchronized;

synchronizing a database with said application data; and

synchronizing a first data repository with said application data.

41. The computer readable medium of claim 40, wherein:

said application data comprises at least one XML file.

42. The computer readable medium of claim 40, wherein:

said application data comprises application data files adapted for transmission over HTTP.

43. A method for synchronizing application data to a cluster of servers, comprising:

notifying a proxy data repository of application data, said proxy data repository running in an enterprise application on an administration server;

sending said application data to said proxy data repository;

forwarding said application data from said proxy data repository to an enterprise application on a server of a cluster of servers;

synchronizing said enterprise application on said cluster server with said application data; and

performing said forwarding and synchronizing steps for all remaining servers in said cluster of servers.

44. The method of claim 43, wherein:

said proxy data repository is subscribed to receive said application data.

45. The method of claim 43, wherein:

said application data comprises at least one XML file.

46. The method of claim 43, wherein:

said application data comprises application data files adapted for transmission over HTTP.

47. The method of claim 43, wherein:

said method is performed as part of a testing process in preparation for deployment of said application data to a production server.

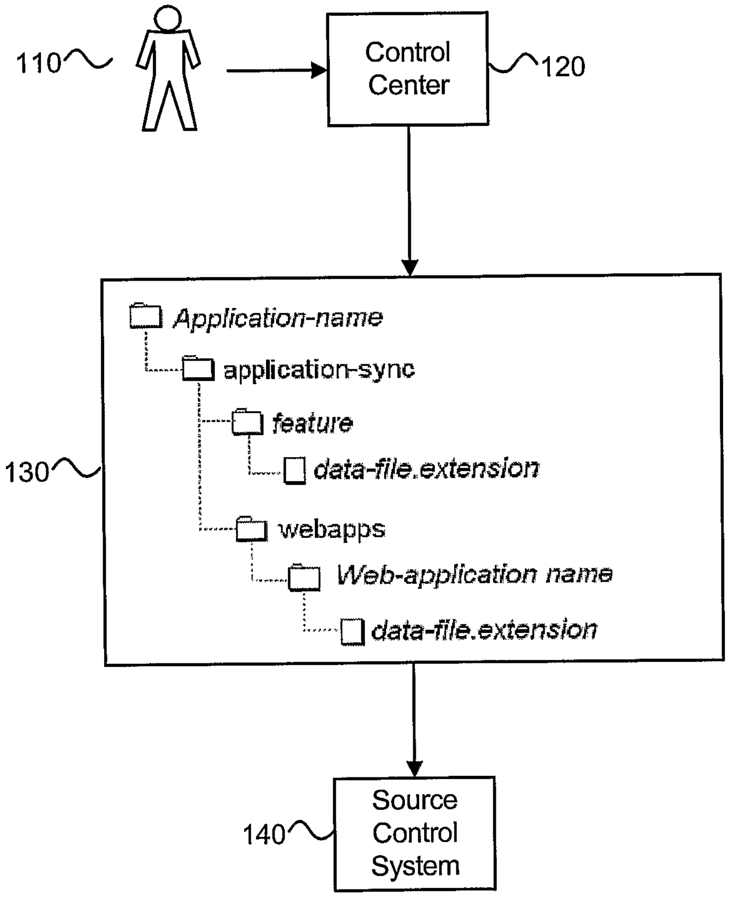

110
Control Center
120
Application-name
application-sync
feature
data-file.extension
webapps
Web-application name
data-file.extension
130
140
Source Control System

Figure 1

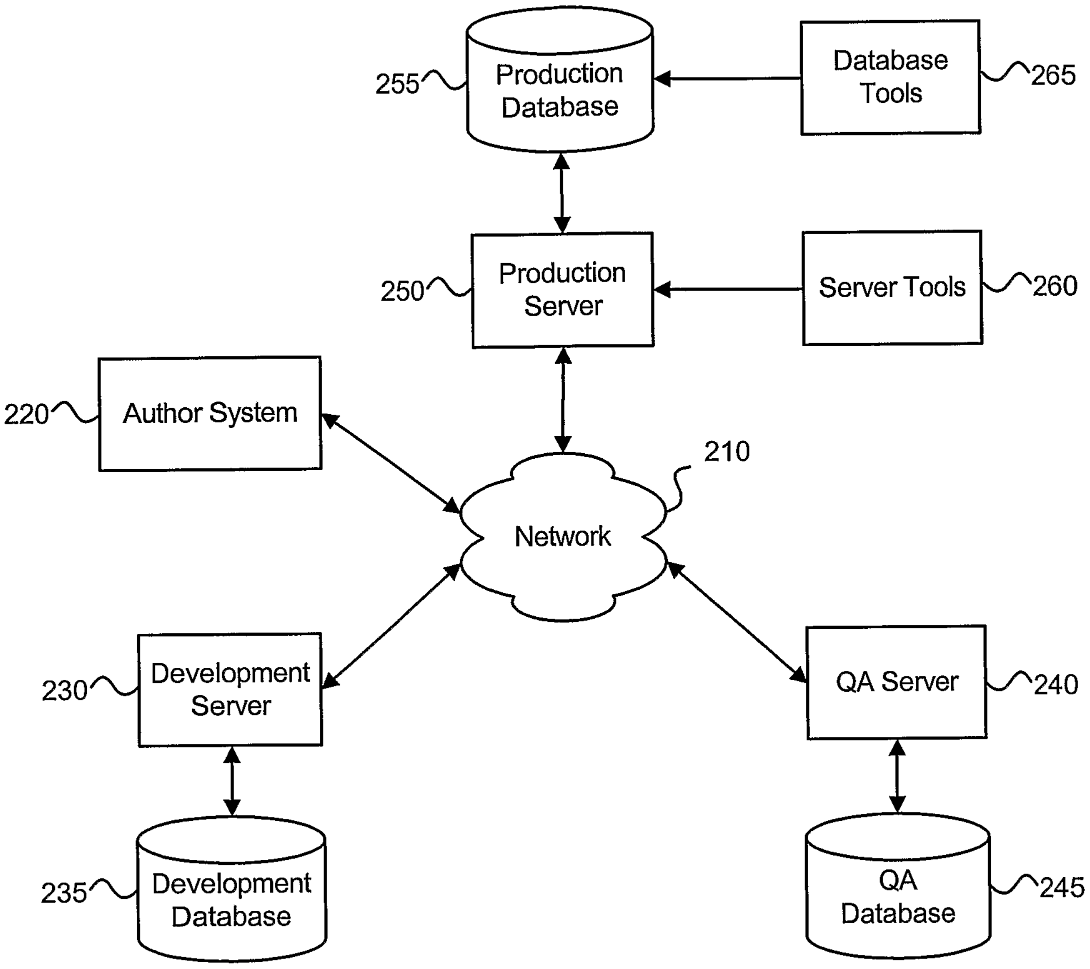

255
Production Database
Database Tools
265
250
Production Server
Server Tools
260
220
Author System
210
Network
230
Development Server
QA Server
240
235
Development Database
QA Database
245

Figure 2

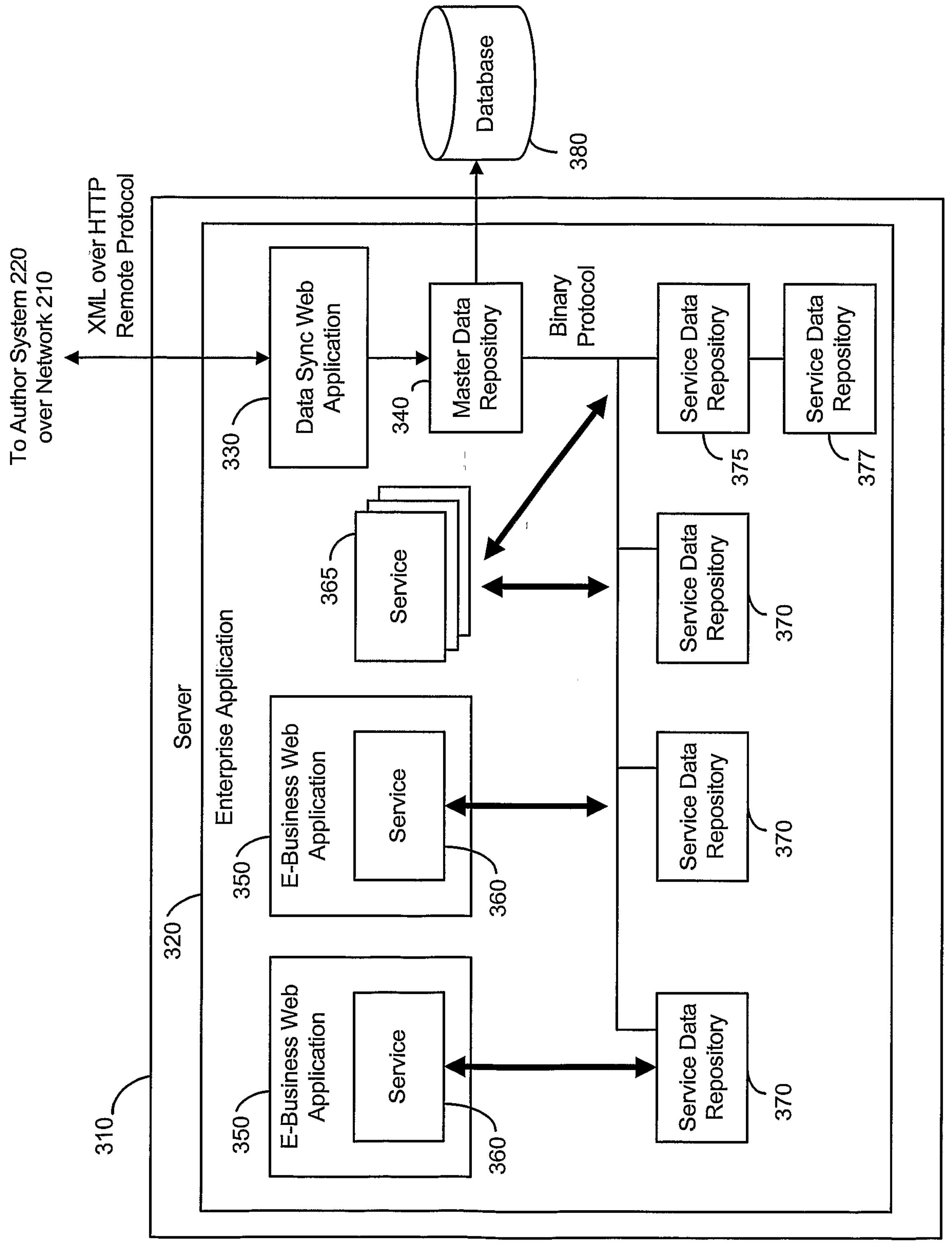

To Author System 220
over Network 210
XML over HTTP
Remote Protocol
Database
380
Server
310
Enterprise Application
320
Data Sync Web
Application
330
Master Data
Repository
340
Binary
Protocol
Service Data
Repository
375
Service Data
Repository
377
Service
365
Service Data
Repository
370
E-Business Web
Application
350
Service
360
Service Data
Repository
370
E-Business Web
Application
350
Service
360
Service Data
Repository
370

Figure 3

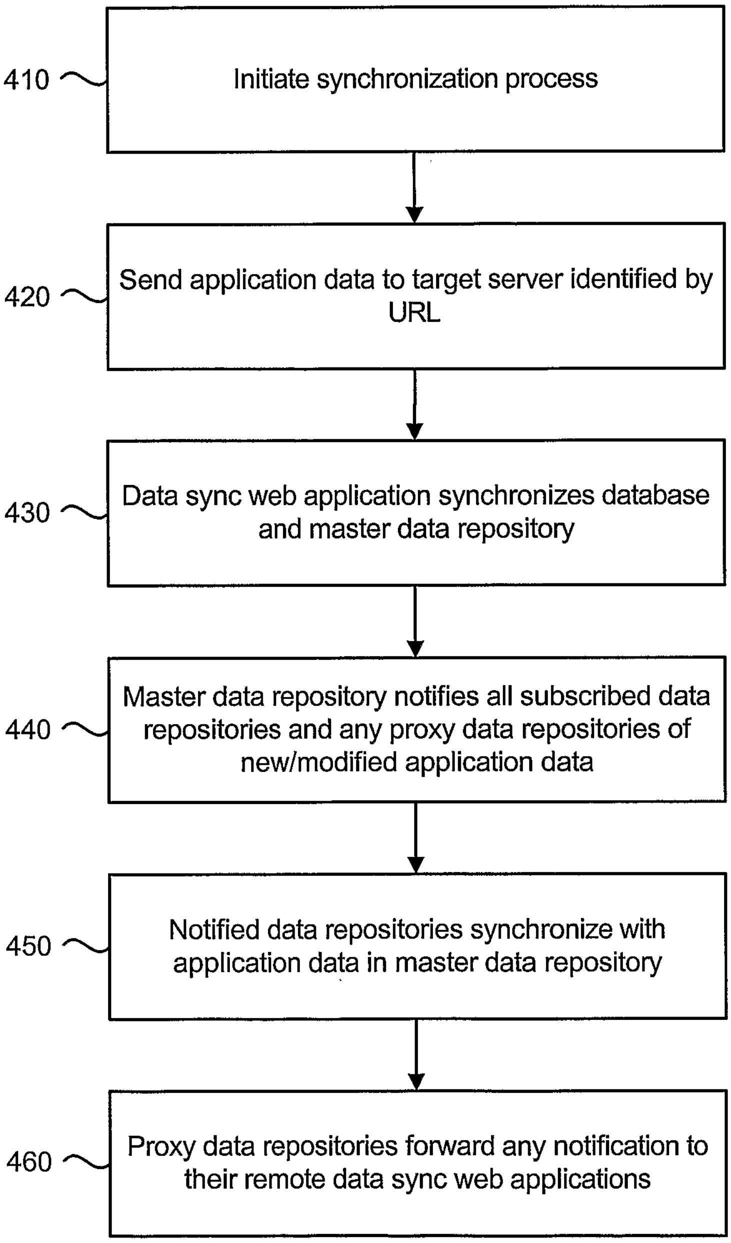

410
Initiate synchronization process
420
Send application data to target server identified by URL
430
Data sync web application synchronizes database and master data repository
440
Master data repository notifies all subscribed data repositories and any proxy data repositories of new/modified application data
450
Notified data repositories synchronize with application data in master data repository
460
Proxy data repositories forward any notification to their remote data sync web applications

Figure 4

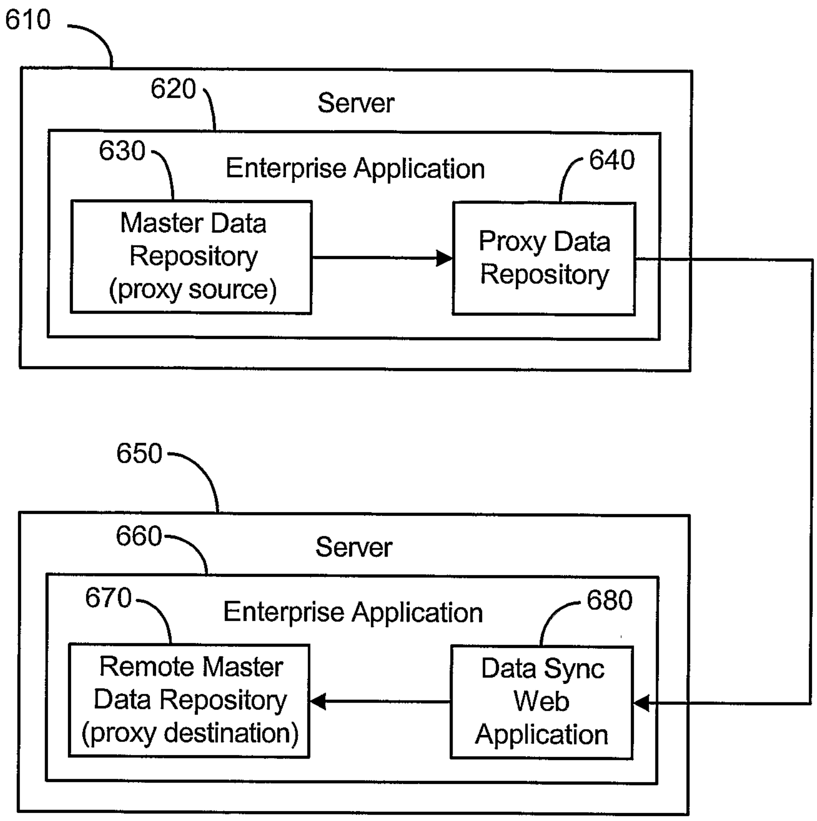

610
Server
620
Enterprise Application
630
Master Data Repository (proxy source)
640
Proxy Data Repository
650
Server
660
Enterprise Application
670
Remote Master Data Repository (proxy destination)
680
Data Sync Web Application

Figure 5

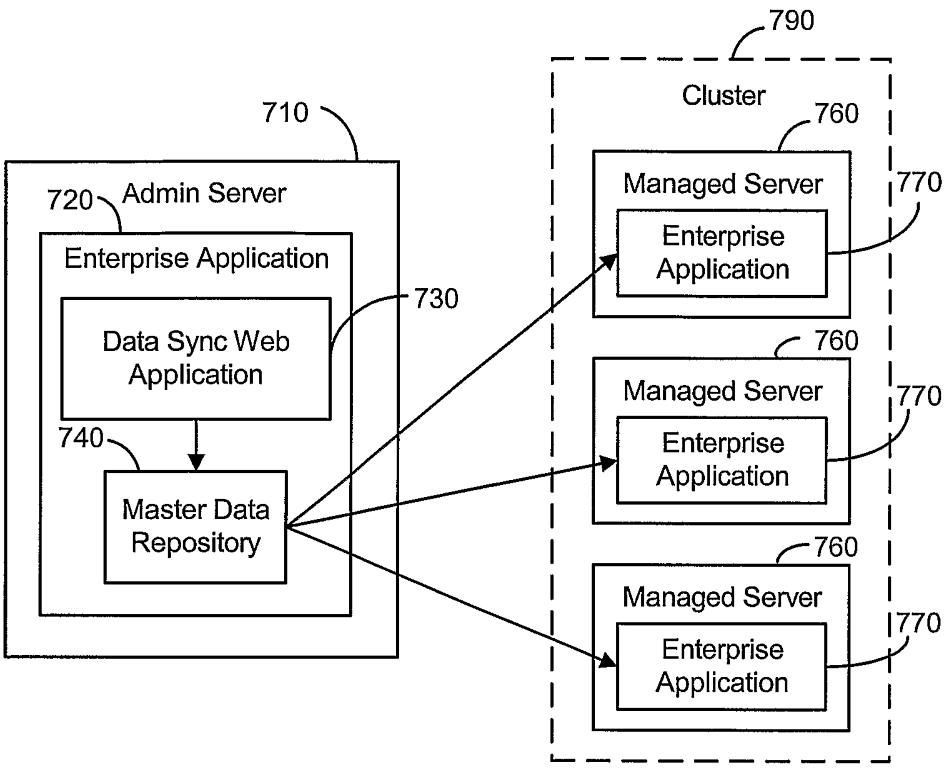

790
Cluster
710
Admin Server
720
Enterprise Application
Data Sync Web Application
730
740
Master Data Repository
760
Managed Server
Enterprise Application
770
760
Managed Server
Enterprise Application
770
760
Managed Server
Enterprise Application
770

Figure 6

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US02/34309

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :GO6F 17/30, 12/00, 15/80, 15/16; HO4M 3/42; H04L 12/28
US CL :707/201, 8, 10; 709/248
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)

U.S. : 707/201, 8, 10; 709/248

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)

WEST, EAST, IEEE, ACM, NPL

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,758,355 A (BUCHANAN) 26 May 1998, col. 3, lines 29-52.	1-11, 15-16, 19-22, 25-26, 28-30, 32-34, 37, 40, 43-44, 47
Y	US 6,092,083 A (BRODERSEN et al.) 18 July 2000, col. 2, line 37-col. 4, line 7.	1-47
Y	US 6,169,794 B1 (OSHIMI et al) 02 January 2001, col. 2, line 3 col. 3, line 37.	1-47
Y	US 5,649,195 A (SCOTT et al) 15 July 1997, col. 2, line 20-col. 4, line 23.	1-47
Y,P	US 6,438,563 B1 (KAWAGOE) 20 August 2002, col. 3, line 44-col. 5, line 33.	1-47

☒ Further documents are listed in the continuation of Box C. □ See patent family annex.

*	Special categories of cited documents:	"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
"A"	document defining the general state of the art which is not considered to be of particular relevance		
"E"	earlier document published on or after the international filing date	"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
"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)	"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
"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	"&"	document member of the same patent family

Date of the actual completion of the international search 24 JANUARY 2003	Date of mailing of the international search report 14 FEB 2003
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231	Authorized officer For FRANTZ COBY James R. Matthews
Facsimile No. (703) 305-3230	Telephone No. (703) 305-4006

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US02/34309

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,838,909 A (ROY et al) 17 November 1998, col. 5, line 19-col. 7, line 25.	1-47
Y	US 5,956,719 A (KUDO et al) 21 September 1999, col.2, lines 19-47.	1-47
Y	US 6,178,172 B1 (ROCHBERGER) 23 January 2001, col. 7, line 1-col. 8, line 5.	1-47