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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 January 2004 (08.01.2004)

PCT

(10) International Publication Number
WO 2004/003686 A2

(51) International Patent Classification[7]: G06F

(21) International Application Number:
PCT/US2003/020046

(22) International Filing Date: 25 June 2003 (25.06.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/392,144 27 June 2002 (27.06.2002) US
10/212,303 5 August 2002 (05.08.2002) US

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

(72) Inventors: **GARIMELLA, Sandilya**; 1712 Via Flores, San Jose, CA 95132 (US). **DALAL, Sanjay**; 575 E. Remington Drive #12M, Sunnyvale, CA 94087 (US).

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

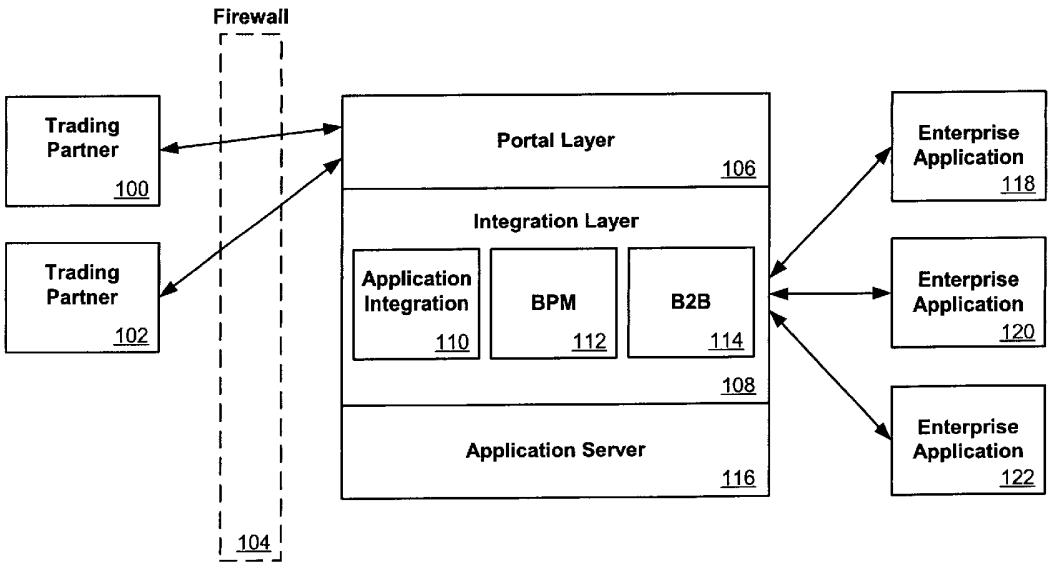
(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, 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, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

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

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.

(54) Title: SINGLE SYSTEM USER IDENTITY



(57) Abstract: When an external user such as a trading partner (100, 102) makes a request into an access point of an application on an application server (116), that external user can be authenticated as a valid user on the system. The identity of the external user can then be switched to an internal system user identity, such as by pushing new user information on the user stack or by adding internal user context. This internal system user identity allows the user to access resources and applications on the application server (116) that are not available to an external user. The use of this single internal system user identity allows for a single login process that can be used for all resources and applications on the server (116). The use of an internal user also prevents an external user from accessing those resources unless the user is first authenticated through a proper entry point.

WO 2004/003686 A2

SINGLE SYSTEM USER IDENTITY

CLAIM OF PRIORITY

This application claims priority to U.S. Provisional Patent Application No. 60/392,144, filed June 27, 2002, entitled "SINGLE SYSTEM USER IDENTITY", and U.S. Patent Application No. 10/212,303 filed August 5, 2002 entitled "SINGLE SYSTEM USER IDENTITY", incorporated herein by reference.

COPYRIGHT NOTICE

CROSS-REFERENCED CASES

The following applications are cross-referenced and incorporated herein by reference:

U.S. Provisional Application No. 60/392,237 entitled "System and Method for Maintaining Transactional Persistence," by David Wiser et al, filed June 27, 2002.

U.S. Provisional Application No. 60/376,906 entitled "Collaborative Business Plug-in Famework," by Mike Blevins, filed May 1, 2002.

U.S. Provisional Application No. 60/377,157 entitled "System and Method for Collaborative Business Plug-ins," by Mike Blevins, filed May 2, 2002.

U.S. Provisional Application No. 60/347,919 entitled "Application View," by Mitch Upton et al., filed October 18, 2001.

FIELD OF THE INVENTION

The present invention relates generally to data security and user authentication.

BACKGROUND

In many e-business systems, businesses would like to maintain tight control over which people have access to sensitive information, such as sales, product, or customer information in a legacy database. In conventional systems this is not an issue, as anyone given access to a system has access to all resources and data on that system. In an enterprise system, any applications can be placed behind a firewall. Certain people are given access to the system through the firewall, but again have access to all resources once inside the firewall. This requires a system administrator to keep a close watch on who is accessing these resources.

Existing e-business solutions for managing business workflow, as well as for enabling standardized business-to-business (B2B) messaging, utilize separate system user identities for each of these applications. A user identity for a business process management (BPM) component provides a user with access to all BPM resources, such as JDBC (Java Database Connectivity) and Enterprise JavaBeans (EJBs). A user identity for B2B provides a B2B user with access to B2B resources, such as messaging resources with possible exceptions such as servlets and JavaServer Pages (JSPs). Administrators for systems using both of these applications have to manage these separate identities. This does not provide for ease of use.

Systems using such applications can have problems with unauthorized users accessing the system using one of these user identities. For example, an unauthorized user could access system data through a system node if that unauthorized user obtained a proper username and password. The unauthorized user could simply generate a request that appears to the system to be an authenticated request.

Some systems have addressed such security concerns by "locking down" the system. For instance, certain systems include an additional access code or flag that provides the system with the ability to allow or disallow Java naming and directory interface (JNDI) lookups. This is a potential problem point, however, as systems may operate in a cluster with more than one node. One of these nodes may want to do a JNDI lookup on another node, which will not be possible if JNDI lookups are locked down. It also will be impossible to use other features of the system. Many system components and resources are interrelated, and lockdown will cause problems for many of these components.

BRIEF SUMMARY

Systems and methods in accordance with the present invention utilize a single system user identity to provide a user with access to resources and applications on an application server. There can be several applications running on an application server, with each application having at least one access mechanism through which an external user can access the application and/or application server. A validation mechanism can be used to validate an external user, such as by comparing information provided by the user against user information in a database. Once an external user is validated, the validation mechanism can switch the identity of the external user to an internal system user identity. Once the user is switched to an internal system user, that user can access any application and/or resource on the application server to which an internal user is granted access. The identity of the user can be switched by pushing internal user information on the user stack for the external user, or by adding internal user context to the external user identity, for example. This switch can be done at any appropriate time, such as when an external user is first validated or when an external user first attempts to access a resource or application requiring an internal user identity.

Other features, aspects, and objects of the invention can be obtained from a review of the specification, the figures, and the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a diagram of a system in accordance with one embodiment of the present invention.

Figure 2 is flowchart for a method that can be used with the system of Figure 1.

Figure 3 is a diagram of a system in accordance with another embodiment of the present invention.

DETAILED DESCRIPTION

An integration application can be built, or layered, on top of an application server **116**, as shown in **Figure 1**. Such an integration layer **108** can consist of a number of applications or components, such as may include a business-to-business (B2B) component **114**, a business process management (BPM) component **112**, and an application integration (AI) component **110**. Other components, such as eXtreme Programming (XP) components, can also be included in the integration layer. XP is a relatively new business standard approach to rapidly developing high-quality, high-value software for customers.

Trading partners **100**, **102** can gain access to the system through access points, such as may be contained in a portal layer **106** built on top of the integration layer **108**. Once a trading partner **100**, **102** is authenticated, that trading partner or user can make requests into enterprise applications **118**, **120**, **122**, for example, through the integration layer **108**.

In order to process such a request, the integration components can communicate with each other. **Figure 2** shows one example of a communication between an application integration component **110**, a business

process management component **112**, and a business-to-business component **114**. A trading partner **200** can send a request to the system that is received by a B2B component **114**. The B2B component **114** can direct the request to the appropriate business processes in order to process the request. A BPM component **112** can manage the workflow for the request. From an external process **202**, the BPM component can make a call to invoke an enterprise information system (EIS) **204**. The call to invoke EIS **204** can pass the request to an application integration (AI) component **110**. The AI component can contain an application view **110**, which provides access to an event connector **210** for the EIS database or datastore **212**. Once the request is processed in the database **212**, a service connector **214** passes the response back through the application view **208** to the BPM component **112**, which is waiting for a response **206**. In the case of asynchronous messaging, the BPM component may not be waiting for a response, but can retrieve the response later, after the presence of the response is detected.

Once the BPM component gets the response, the response can be passed back to the external process **202** and then to the trading partner **200** through the B2B component **114**. In prior systems, the user would need a valid username and password for each of the B2B **114**, BPM **112**, and AI **110** components.

As trading partners often make requests into the system, it is desirable to limit their access to only those resources in the system which they might need, and to which their access is desired by the business or entity owning the data or resources. In processing these requests, a system administrator may not wish to give trading partners any additional capabilities or resource access. Simply because these users may be valid users on the system, and may each have a valid user identity, does not mean that the trading partners should have access to everything on the system. One way to control user access is to only

provide System users with access to certain resources, and not simply any valid users on the system.

A system and method in accordance with one embodiment of the present invention uses a single system user identity that provides access to all these integration components, as well as any associated resources or objects. As shown in **Figure 3**, the use of a single system identity to authenticate a user can allow components, such as BPM **312** and B2B **310** components, to communicate without having separate logins or user authentications. Components can have multiple access points **306**, **308**, such as a series of transport servlets that allow a request to be transported into the system. Access to the B2B component can also be obtained through a BPM component. Through BPM, a user can send a message to B2B using a B2B plugin or B2B interface, for example. Another access point could be a B2B console that provides for user login.

Since a single system user identity allows a user to accomplish tasks such as making requests against a system database and acting on a B2B depository, systems and methods in accordance with the present invention secure the access points for B2B, BPM, and potentially any other integration component to avoid the processing of unauthenticated requests. For example, a B2B user can be authenticated when that user enters through one of the B2B access points. Once the user is authenticated at one of these access points, such as by verifying the username and password information provided by the user against a table in a database, the identity of the user can be "switched" to an internal user or system user. This switched user shall be identified from this point forward as simply a "System" user.

The use of a single system user identity can be advantageous, as components such as B2B and BPM components can have, or provide access to, a number of resources. A message can be required to access resources for both these components. These resources can include, for example, databases, queues and administrative frameworks using MBeans. As a

message travels through these components, the identity of the user initiating that message is propagated with the message. Additional identity information can also be propagated with the message, which can be referred to as System user information. This System user information allows a user to have access to any and/or all of the resources which these components provide, where individual component user identities may only provide access to resources for the respective component.

The use of a single system user identity also means that system administrators do not have to configure all component resources for each user that may be accessing these systems. For external access, all that may need to be configured are the appropriate policies that allow a user to be verified and enter the system. After a user passes through an access point and is verified, the system can act on behalf of that user by attaching a System user identity to that user. This approach can provide protection throughout the entire application server system, including components such as B2B and BPM.

An entire runtime system can be controlled under a single user identity. For each valid incoming user, that incoming user can be required to be switched to a System user before the system will process the request. For example, if a valid trading partner comes into the system with the username of "UserA", UserA will be a valid username on a given application server. UserA can gain system access through any system entry point. If UserA comes through a proper entry point, UserA can be authenticated and switched to an internal user identifier, such as System. Once this switch is completed, the user will appear to the system as System, instead of UserA. The System user can be given access to specified system resources that are not available to UserA. From this point forward the external user will be referred to as UserA. References "System" and "UserA" are used for convenience and demonstration only, and are not intended to limit the possible designation or naming schemes that can be used for internal and/or external users.

One advantage of a single system user identity is that an application server can provide access control that is well-defined. Even though UserA may be a valid user on the system, access to any resource on a machine can be limited to System users. UserA can still be granted access to certain resources without the switch, but may not be able to access a critical resource, such as a database resource. A user logged into the system as an external user cannot then access certain controlled resources.

The switching of a user identity from an "external" user to an "internal" user can be much more than simply a transformation of the username. As a user enters an entry point, or access point, that user can be authenticated and another user can be pushed on top of that user. This switch results in resource access being granted to this "new" user. The pushing of a new user on top of the existing user prevents an unauthorized user, having obtained a valid username and password, from coming through a specified entry point and doing a JNDI lookup. A JNDI lookup allows a user to lookup a resource on the application server. If a system does not require a system user to be pushed on top of an external user in order to do a lookup, that external user could access the resource directly from any entry point or node in the system. When an available thread is selected for a message, context can be loaded for the thread which includes the a user stack in the thread address space. The "new" or additional user information can be pushed on top of this user stack to identify the source of the thread of execution.

The user can therefore be switched by creating an authenticated user context for valid users on a platform. Each time a resource or component is to be accessed for a particular user, this context can be pushed on the user stack, and the new user identity can be assumed from that point forward. It can be beneficial, for security reasons, to be able to lock down the server for a given internal user. Even though each system component can have a set of valid users, it can be more convenient and can offer more control to utilize a single system user identity to access resources across the system.

Even if a single system user identity does not provide any additional protection for every system component, an internal username can provide access control for an integration layer. Even if someone knows a valid username and password on the application server, this valid username and password will not necessarily grant access to the integration layer. At any integration entry point, an error can be thrown and a connection closed for an external user request, as only internal users are granted access to internal integration resources. Any external access can be prohibited.

Entry points

External users can be prevented from sidestepping an entry point, such as a portal, and taking advantage of other channels to access application server resources. These users should not be allowed to have read/write access to information in a database, for example, unless they are first authenticated as having those privileges. As the entire runtime system can run on a single user, there is no need for multiple authentications.

One entry point that can be used for a B2B component is a transport servlet. A transport servlet can be configured to receive a message from across a network and process that message. Other access points can include, for example, BPM studios and system user interfaces, which can each include a user login screen. When users come in through these tools, or access points, the users can have access to a limited set of tasks. As mentioned, a BPM studio can allow a user to login using a valid username and password. This studio can be implemented as a tool that allows a user to create a workflow process, or to select an exsisting process. User authentication can be done in a studio, before the user is switched to an internal user.

Once a single system user identity is implemented for various integration components on an application server, there is the possibility of a message coming into a worklist or studio, typically a BPM task, and actually generating a B2B message. A message coming into B2B can itself trigger a workflow, for example, and many other inter-component exchanges are

possible. Additional access controls can be defined to account for these inter-component exchanges. For example, there can be a set of permission groups in BPM that can indicate whether a user can create a template, process a workflow, delete a template, or monitor instances. A valid system user can be defined as a member of this group, which can then have access to all tasks in the set.

In order for a studio to work with these templates, it can first be determined whether a user is part of this group. This can be configured automatically so that the user gets all privileges to the set of tasks. If that user wants to create a template from the database, however, that user may need additional privileges. Until the user calls a runtime service where the switch is done, for example, the user is still external user UserA. These extra privileges can be given to users directly, which can provide undesirable exposure, or the extra privileges can be provided to internal users which have undergone the switch to an internal System user.

Not every user needs to be automatically switched when entering through an access point, or entry point. For instance, if a user wants to create a workflow template, and that user belongs to a group that has the privilege to create a workflow template, there is no need to push an internal identifier on that user before that user accesses the appropriate bean or resource. If a call gets all the way to this resource, a check can have already been done to ensure the incoming user can do that task. The request can assume an identity that has higher privileges than those provided by the transport servlet, such as requests that are coming over the network using a secure sockets layer protocol (SSL) for authentication. The certificate that a website uses to make the SSL request can be mapped to a user, so the request can be certificate-based instead of password-based. For users verified by SSL, there may not be a need to push additional authentication.

EJBs

In order to perform various tasks, a system can utilize a set of EJBs, as well as a common repository, such as may utilize JDBC. Once authenticated, a user can choose a task such as "create workflow." This task can be accomplished through a call to an EJB. Each EJB, as well as the associated deployment descriptor, can have access to, or can contain, the system identities. This allows the EJBs to process requests only for System users. Once a user is inside an EJB and executing a task such as accessing the database to store and retrieve information, the username can actually be the internal System username.

Integration Components

In an e-business environment, collaboration between trading partners can occur through the exchange of business messages that contain XML or nonXML documents in a secure, choreographed arrangement called a conversation. Access to the conversation, as well as conversation management, can be provided by a business-to-business (B2B) component. A conversation is, quite simply, a series of business messages exchanged between trading partners, the composition of business messages and the sequence of an exchange being handled by collaborative or public business processes. The composition and sequence of messages can also be handled by Java messaging applications. Conversations can be complex and long-running, or they can be short-lived. Each conversation can have a unique name, and each participant in a conversation can have a conversation role, such as that of a buyer or a supplier in a supply-chain arrangement.

Details of a conversation, including its name and version, the roles of the participants, and the business protocols it uses, can be specified in a conversation definition. Integration specialists can create conversation definitions and monitor running conversations using a console, for example, that is provided by a B2B component.

Business processes can be designed to be started or stopped by users, or to include tasks that must be performed by users. These tasks can include making discretionary decisions, handling exceptions, or troubleshooting problems. An application integration component can provide an application called a "worklist" that people can use to start and stop processes, as well as to interact with a running process. Using the worklist, users can handle business process tasks assigned to them, such as making a decision about a customer's credit limit, or they can respond to messages from a process.

An e-commerce community can be formed when a trading partner joins other trading partners to pursue a common business objective. An e-commerce community can exist in different forms, and for different purposes. It might, for example, span multiple departments within a company to manage inventory across the company. A community can also span multiple companies across firewalls and over the Internet to manage a supply chain or a multi-step purchasing arrangement, and can include trading partners both within a company and in other companies, such that one or more trading partners interact with trading partners in other companies.

To participate in the conversations of an e-commerce community, integration specialists can use a B2B console to configure trading partners. Specifically, the specialists can assign trading partners the names by which they will be known in the conversation, and can specify the delivery channels to be used for the exchange of business messages.

A B2B component can also provide certain security services, which can be built upon security services provided by the underlying application server. These security services can include features such as an SSL-based secure platform for conversations, certificate verification that can be used to authenticate the identities of trading partners, digital signatures that can be attached to business messages being exchanged by trading partners, support for nonrepudiation of origin and nonrepudiation of receipt, which are often required by law for critical business messages, and data encryption for

business protocols that require this support. B2B integration can be used to quickly and easily connect enterprises, to create and execute collaborative trading partner agreements, and to support multiple business protocols (cXML, ebXML, XOCP, RosettaNet, etc.).

An application integration (AI) component can utilize J2EE CA-compliant adapters, such as service and event adapters, to connect to an EDI-capable system. An application view can be used to integrate business processes with the EDI system. The application integration component can provide the functionality needed to design, execute, and monitor complex, enterprise-wide processes that span applications, systems, and people. The AI component can include a Java-based process engine that manages the run-time execution of business processes throughout the enterprise.

The foregoing description of preferred embodiments of the present invention has been provided for the purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations will be apparent to one of ordinary skill in the relevant arts. The embodiments were chosen and described in order to best explain the principles of the invention and its practical application, thereby enabling others skilled in the art to understand the invention for various embodiments and with various modifications that are suited to the particular use contemplated. It is intended that the scope of the invention be defined by the claims and their equivalence.

CLAIMS

What is claimed is:

1. A system for validating a user on an application server, comprising:

an application server;

at least one application running on the application server, each application having an access mechanism through which an external user can access at least one of the application and application server; and

a validation mechanism for validating an external user gaining access through an access mechanism, the validation mechanism switching the identity of a validated external user to an internal user identity recognized by said at least one application running on the application server.

2. A system according to claim 1, wherein:

said at least one application is an integration application.

3. A system according to claim 1, further comprising:

at least one trading partner having permissions on the application server and any applications running on the application server.

4. A system according to claim 1, further comprising:

a database in communication with the application server for storing information related to any user of an application.

5. A system according to claim 1, wherein:

the at least one application running on the application server has an access mechanism that is a portal component.

6. A system according to claim 1, wherein:

said validation mechanism switches the identity of the user, the user identity being a user stack, by pushing internal user information on the user

stack.

7. A system according to claim 1, wherein:

said validation mechanism switches the identity of the user by adding internal user context to the eternal user identity.

8. A system according to claim 1, further comprising:

application resources that are accessible only to a user with an internal user identity.

9. A system according to claim 1, wherein:

said validation mechanism switches the identity of the user to the only internal user identity recognized by each application running on the application server.

10. A system according to claim 1, wherein:

each application in said at least one application can communicate with any other application running on the application server without re-validating the external user.

11. A system according to claim 1, wherein:

the at least one application has multiple access mechanisms.

12. A system according to claim 1, wherein:

the at least one application has an access mechanism selected from the group consisting of databases, queues, and administrative frameworks.

13. A system according to claim 1, further comprising:

application resources that can be accessed by an external user without the identity of the external user being switched.

14. A system according to claim 1, wherein:

said validation mechanism switches the identity of a validated external user only after the external user attempts access requiring an internal user identity.

15. A system according to claim 1, further comprising:

a plurality of applications, wherein the internal user identity is recognized by each application of said plurality of applications.

16. A method for validating a user on an application server, comprising:

receiving a request from an external user to an access point of an application on an application server;

authenticating the external user; and

switching the identity of the external user to an internal user identity, the internal user identity providing access to resources for any application running on the application server.

17. A method according to claim 16, wherein:

authenticating the user involves checking information for the external user against user information in a database in communication with the application server.

18. A method according to claim 16, wherein:

switching the identity of the external user to an internal user identity involves pushing internal user information on a user stack for the external user.

19. A method according to claim 16, wherein:

switching the identity of the external user to an internal user identity involves adding internal user context information to the external user identity.

20. A method according to claim 16, further comprising:

limiting access for application resources to users with an internal user identity.

21. A method according to claim 16, further comprising:

selecting a single internal user identity to be used to provide access for each application and resource on the application server.

22. A method according to claim 16, further comprising:

allowing an external user to access certain resources on the application server without switching the identity of the external user.

23. A method according to claim 16, wherein:

the identity of the external user is switched only after the external users attempts access requiring an internal user identity.

24. A computer-readable medium, comprising:

means for receiving a request from an external user to an access point of an application on an application server;

means for authenticating the external user; and

means for switching the identity of the external user to an internal user identity, the internal user identity providing access to resources for any application running on the application server.

25. A computer program product for execution by a server computer for validating a user on an application server, comprising:

computer code that can receive a request from an external user to an access point of an application on an application server;

computer code that can authenticate the external user; and

computer code that can switch the identity of the external user to an

internal user identity, the internal user identity providing access to resources for any application running on the application server.

26. A system for validating a user on an application server, comprising:

means for receiving a request from an external user to an access point of an application on an application server;

means for authenticating the external user; and

means for switching the identity of the external user to an internal user identity, the internal user identity providing access to resources for any application running on the application server.

27. A computer system comprising:

a processor;

object code executed by said processor, said object code configured to:

receive a request from an external user to an access point of an application on an application server;

authenticate the external user; and

switch the identity of the external user to an internal user identity, the internal user identity providing access to resources for any application running on the application server.

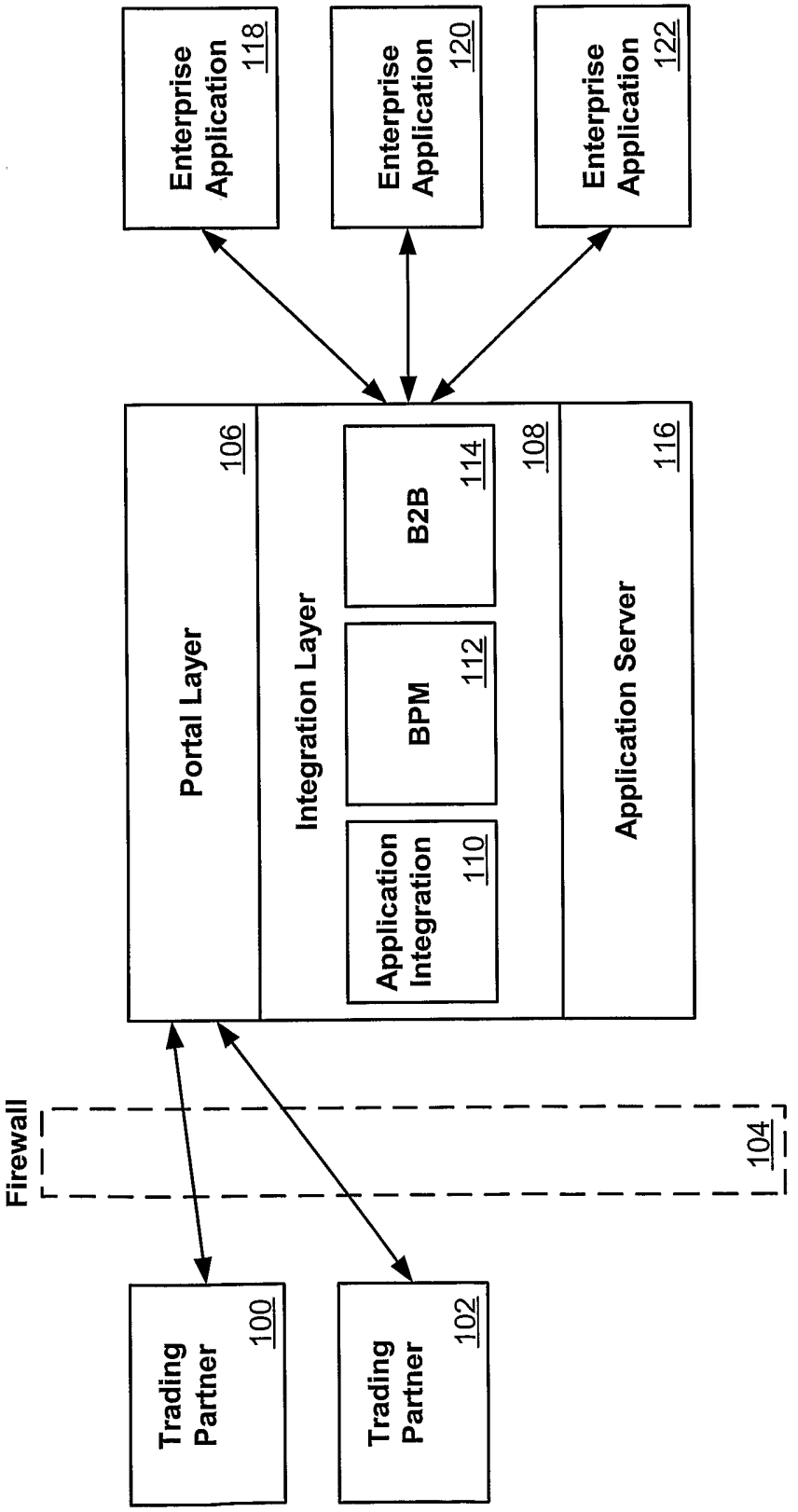

Enterprise Application 118
Enterprise Application 120
Enterprise Application 122
Portal Layer 106
Integration Layer 108
B2B 114
BPM 112
Application Integration 110
Application Server 116
Firewall 104
Trading Partner 100
Trading Partner 102

Figure 1

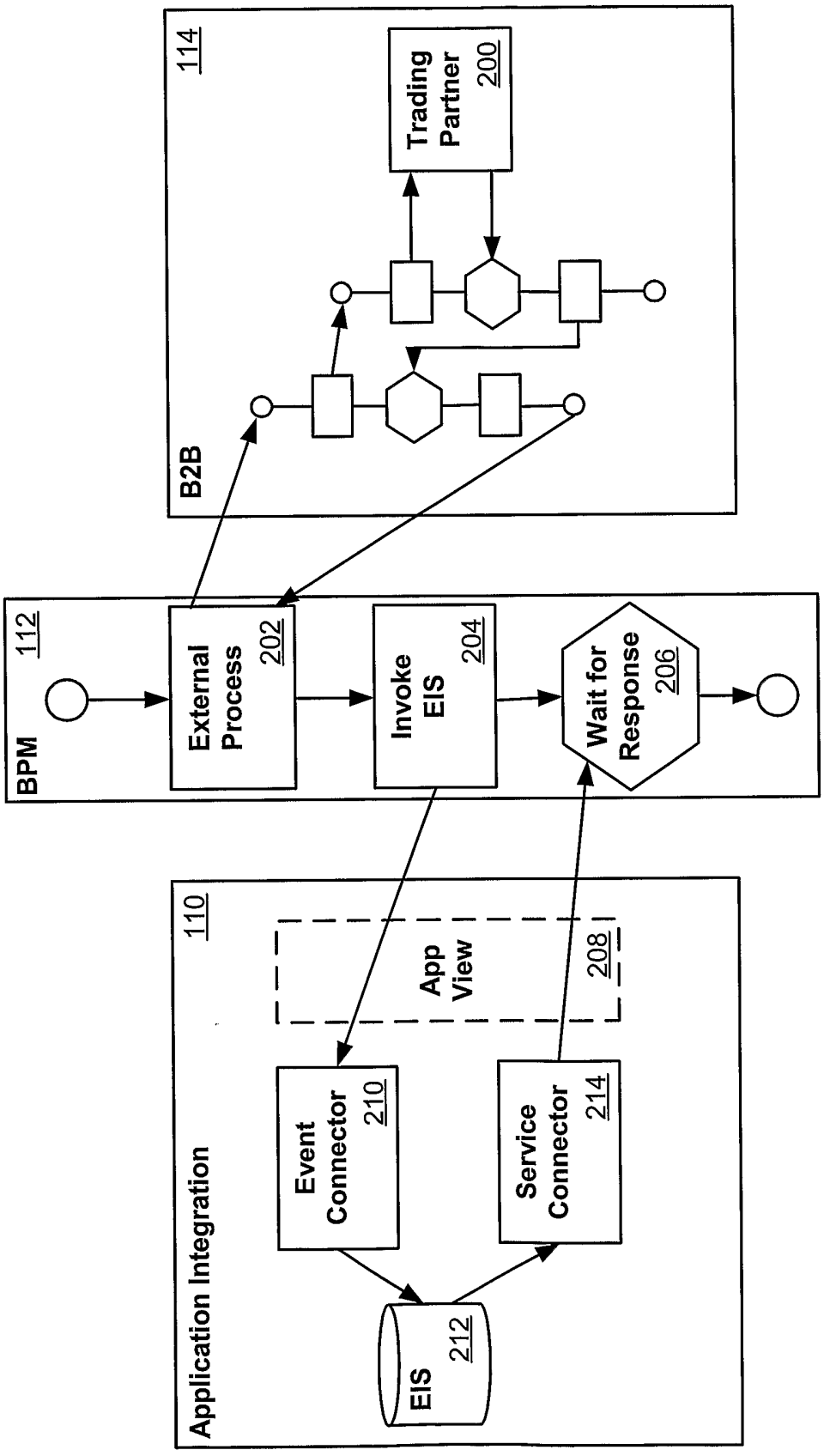

114
B2B
Trading Partner
200
112
BPM
External Process
202
Invoke EIS
204
Wait for Response
206
110
Application Integration
App View
208
Event Connector
210
Service Connector
214
EIS
212

Figure 2

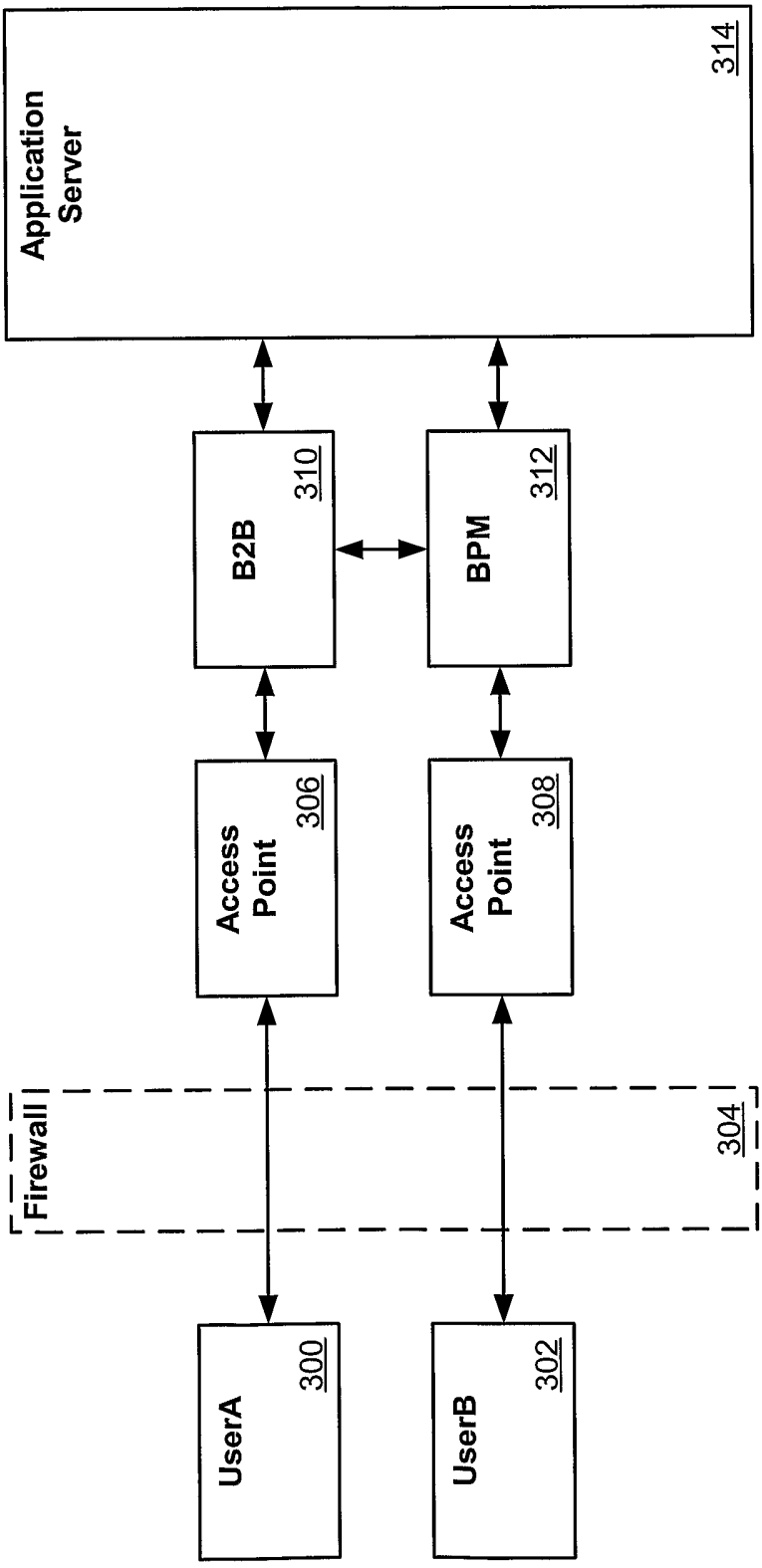

Application Server
314
B2B
310
BPM
312
Access Point
306
Access Point
308
Firewall
304
UserA
300
UserB
302

Figure 3