Technical note

WebEntree: A Web service aggregator

by Y. Zhao

This technical note introduces IBM's WebEntree, a single-log-in Web service aggregator. WebEntree provides an aggregated Web service on top of distributed Web service systems (as components) with a centralized access control and content customization facility. Each service system can have its own access control facility and provide its own independent service. WebEntree implements a flexible and dynamic component-bundling mechanism, and can provide personalized service with user-selected component sets. WebEntree offers a convenient way for new components to be "plugged in" and "played." The owner of the aggregated Web service can keep each component's original branding, add more information, filter out certain content, or customize the presentations. WebEntree also provides a single user registration and authentication interface for all of its user-selectable service components. WebEntree currently accommodates Web service components invoked via HyperText Transfer Protocol (HTTP, i.e., under a Web server) and service components invoked directly from local or remote application programming interfaces. Other component interfaces are planned.

Web service commercialization and personalization usually involve user access control. This means that users must register to obtain a service, and they must be authenticated each time to access the services, e.g., by providing user identification and password. With a growing number of Web services, each with its own authentication facilities, Web users have been inconvenienced. For example, a user's identification (ID) in one system may not be acceptable for another system because of ID conflicts. Also the password validation policy (e.g., time period, character restrictions, etc.) in one system may differ from another. Consequently, a user must have multiple user IDs and passwords, and must be authenticated several times to access different services. Multiple user IDs and passwords are inconvenient to use and difficult to manage. WebEntree solves this problem by providing one common entrance for a user to access all Web service systems, identified as *Web service components* in this note. A user can register and be authenticated only once at entrance; WebEntree then handles user registration and authentication to Web service components. WebEntree also provides the convenience of consolidating the user's Web services in one place, saving exploration time.

In addition, WebEntree assists a Web aggregation service provider by not only providing a single log-in entrance for users, but also the ability to change the Web service components' original branding, customize the presentation, filter out certain content, and add the service provider's own advertisements. Contents from the different Web service components are made accessible by WebEntree prior to sending them to end users, which allows the aggregation service provider to customize the content.

Through WebEntree, the aggregation service provider can also provide personalized services. Customers can select the prebundled Web service component groups they want via initial registration, and they can change their selections at any time. With the use

0018-8670/98/$5.00 © 1998 IBM

of WebEntree's graphical user interface (GUI)-based administration subsystem, an aggregation service provider can add or remove a Web service component easily. Also, the service provider can dynamically bundle the service components into different groups, and manage user information and access to the different groups. The administration activity will not interrupt customer service.

Some WebEntree application scenarios follow:

- *Web service center as a virtual Web site:* WebEntree enables the separation of Web service from Web contents, greatly reducing complexity for the Web service provider. Thus, a Web service provider can create a service Web site without hosting the actual contents, offering a variety of popular Web applications and services from many locations.
- *Existing Web publication service provider enhancements:* A publisher with an existing Web site can add other interesting Web sites. This broadens the contents and colors of the site and adds to its appeal. WebEntree also enables the integration of multiple Web service systems that the service provider may have.
- *Service site in a company's intranet:* In a company's intranet, there may be a need to construct a Web service site that aggregates a variety of Web services from outside as well as inside the company. By using WebEntree, employees do not need to individually pay registration or service fees for the services provided because these items are handled in one place. The single registration and log-in interface and personalized home page also provide employees with access convenience. In addition, the company can customize and filter the Web contents.
- *Service site for an extranet:* An extranet is a network among partners. There may be access control involved with each partner's Web site. WebEntree makes it easier for a user to access the participating Web sites.

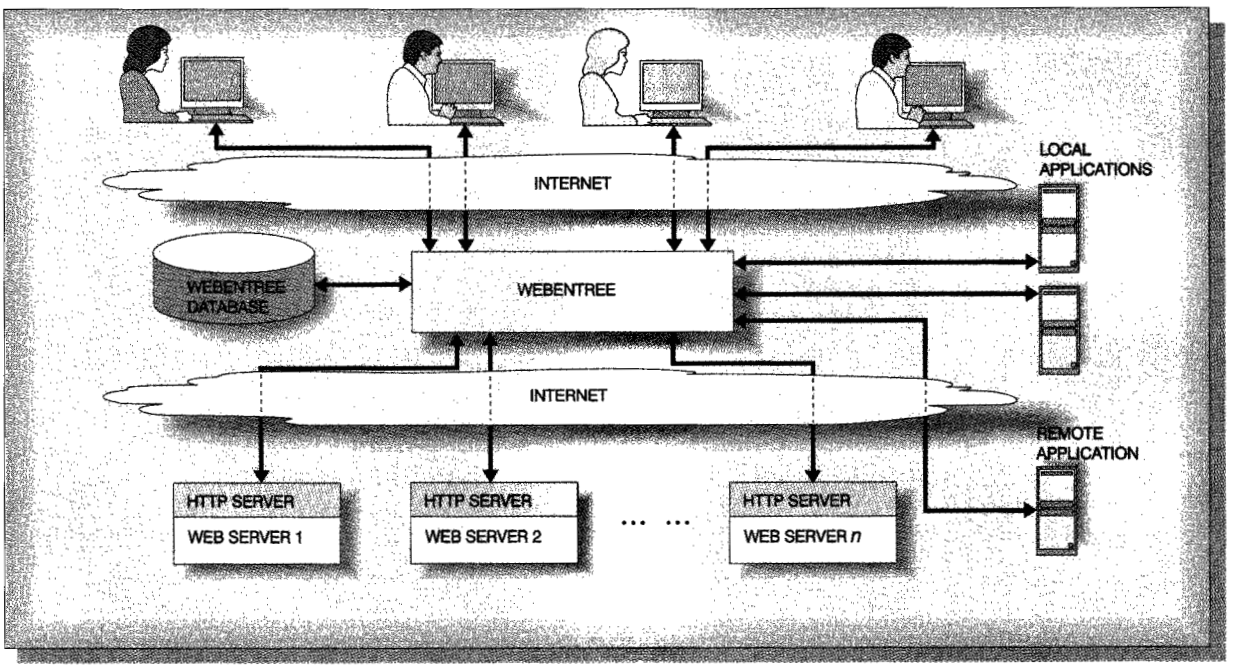
The *Web content* mentioned earlier is information that is representable by a Web browser. It can be compound documents or business data. It can be dynamically generated from database query and computation or from static files. The *Web service* is to provide Web content from a Web server to Web browsers. *Web service aggregation* provides collective Web services from involved Web services systems (or components). It is different from integration, which tends to provide a more tightly coupled system.

Single registration and authentication efforts (or single sign-on) for non-Web environments have been around for a long time. A survey can be found in Hursti.[1] IBM's Global Sign-On product is one example.[2] More can be found in the documents from The Open Group.[3,4] These solutions usually are based on DCE (Distributed Computing Environment) or LAN (local area network) architecture, and for multiple hosts in an affiliated environment. In a Web environment, the most popular service systems are based on HyperText Transfer Protocol (HTTP),[5] or so-called Web servers, although Common Object Request Broker Architecture (CORBA**)-based IIOP (Internet Inter-Orb Protocol) servers are beginning to be considered now as well. These Web service systems usually are operated in a nonaffiliated environment. The problem of single registration and authentication must be solved for Web service systems operated across all different protocols.

In HTTP, the response to a request must come from the server to which the request was sent. This makes it difficult for requests to different HTTP servers to go through a single aggregator gateway. Proxy servers[6] can be a solution. HTTP-based proxy servers have been implemented by Netscape,[7] IBM, and other companies. Services available via proxy servers include firewall, cache,[7] and content filtering and selection.[8] The problem with this solution is that the proxy server must be specified on the Web browser. Most corporate Web clients use a proxy server to go through their firewalls and to get Internet access. Obviously, different proxy servers are used by different corporate firewalls. Since only one proxy server entry is allowed for each Web browser (although proxy server relay is allowed from the entry proxy server), it is difficult for a common service proxy to reach all users inside different firewalls (each firewall proxy has to be configured separately to use the common service proxy). It will be more complicated when multiple common service proxies are involved through different Web service providers. WebEntree has been designed to solve this problem in an alternative way.

WebEntree adopts object-oriented architecture and design. It is based on the object-oriented Java** Web servers (Web servers that support Sun Microsystems' servlet application programming interfaces) and is implemented purely in the Java language; therefore, it is platform-independent.

Figure 1 The conceptual architecture of WebEntree



Architecture

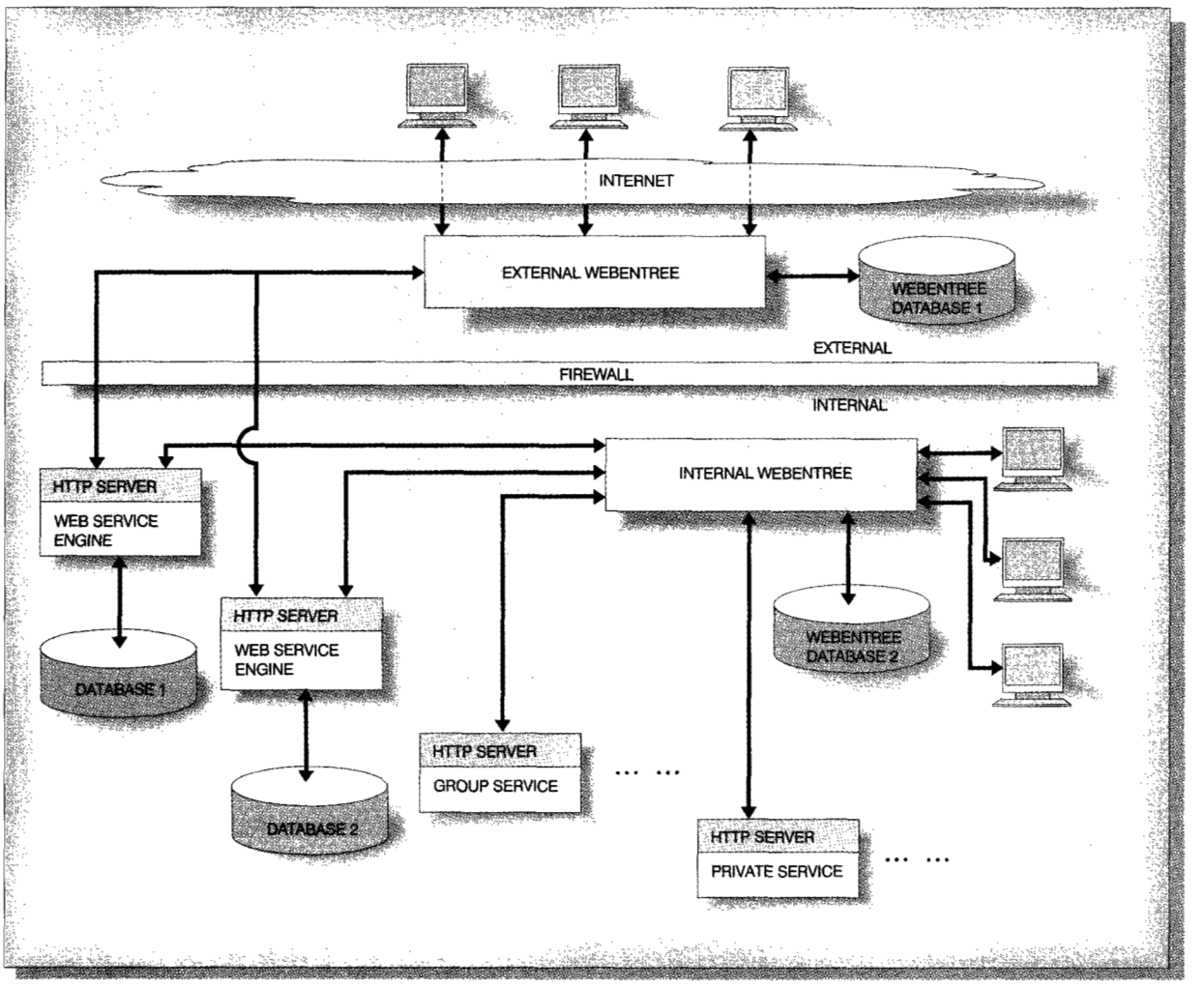
The conceptual architecture of WebEntree is shown in Figure 1. WebEntree can provide services from either inside a firewall (to serve intranet customers) or outside a firewall (to serve certain internal information to the world), as illustrated in Figure 2. To enhance security, digital certification can be used to establish trust between Web clients and WebEntree. This can be implemented by using secure HTTP based on SSL (Secure Socket Layer). Some Web service components may also choose to use SSL with certificates for security enhancement. Most current Web servers support secure HTTP protocol. The challenges in using SSL are (1) the involvement of a third party who issues the certificates and (2) certificate management in both Web client and server.[4] The third party must be trusted by both the other two parties. Depending on the needs of a Web service provider, WebEntree can be configured to use or not to use SSL.

As shown in the system architecture depicted in Figure 3, WebEntree has three main subsystems: user registration, service engine, and administration. The user registration subsystem coordinates user registration information with the corresponding service components. The service engine accepts requests from Web browsers, passes them to the corresponding service components, obtains responses from the components, and sends them back to the Web browsers. Also, the service engine conveys user authentication information to the corresponding components, and provides various service options. The administration subsystem is implemented with a graphical user interface, which provides a convenient mechanism for managing users and Web service components, as well as for service and Web content customization.

A data manager maintains a repository for user, service component, user access group, and component bundling information. It also manages a set of persistent objects to accommodate repository accesses from different threads and processes. This data manager and relevant persistent objects are implemented on top of JDBC (Java Database Connectivity), allowing access to different database products.

A user must be registered to obtain services from WebEntree. User registration information is prop-

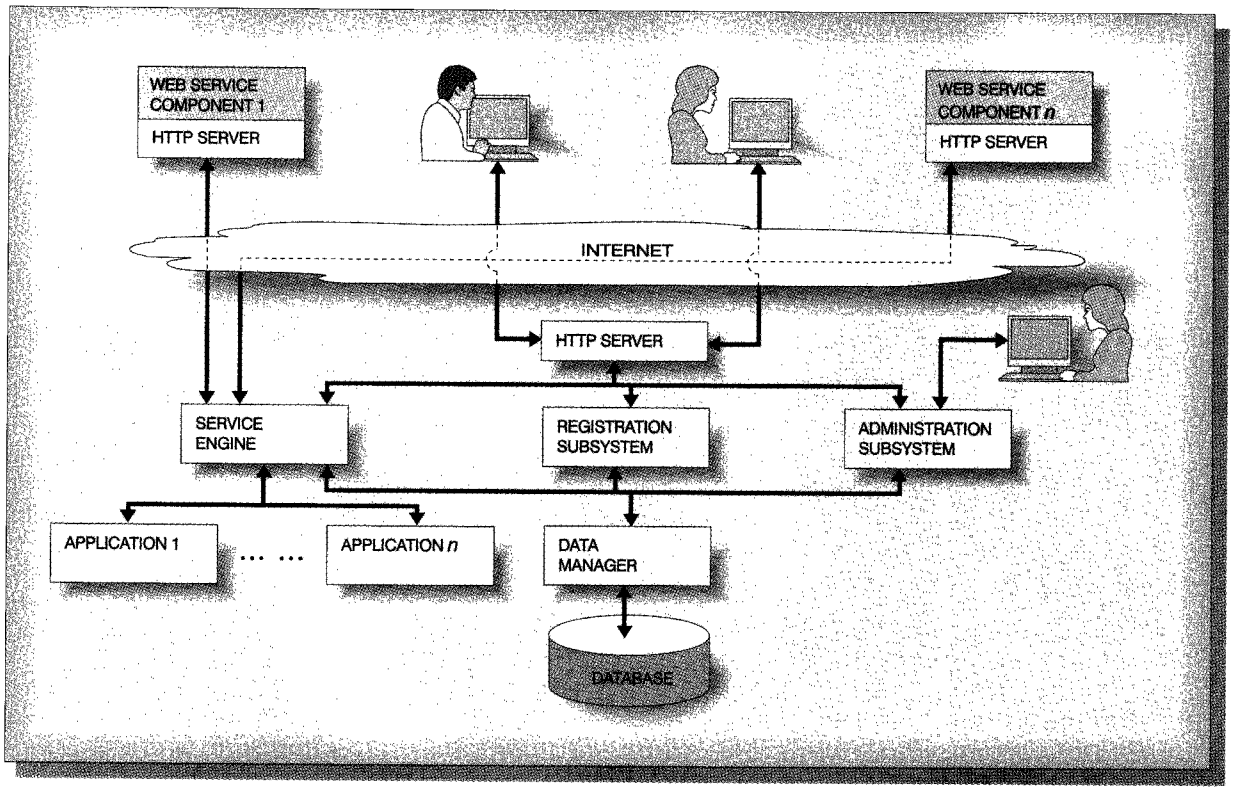
Figure 2 WebEntree for both intranet and extranet Web users



agated to the user-selected Web service components or applications by the registration subsystem. Initial groups of service components can be selected during registration. After registration, a user is granted a user ID, a password, and a certificate if SSL authentication is involved in a selected Web service component. Then, the user can log in to the member service provided by WebEntree. After user log-in, the service engine authenticates the user and provides a personalized home page that includes hyperlinks to all service components the user has selected. When the user clicks on a component's hyperlink, the request is sent to the service engine. The service engine first compares the user's access to the accessibility requirements for the requested service component. If the user has proper access, the service engine will retrieve the Web content from the selected URL (uniform resource locator) and send it to the user. Also, users can post content to a Web service component. Optionally, the service engine can also support usage logs and reporting, and an interface to a billing or payment subsystem.

The WebEntree administration subsystem is implemented in two versions: a server-based Java application version and a Java applets version, viewable with a Web browser. The user interface is basically the same in both versions.

Figure 3 WebEntree's system architecture



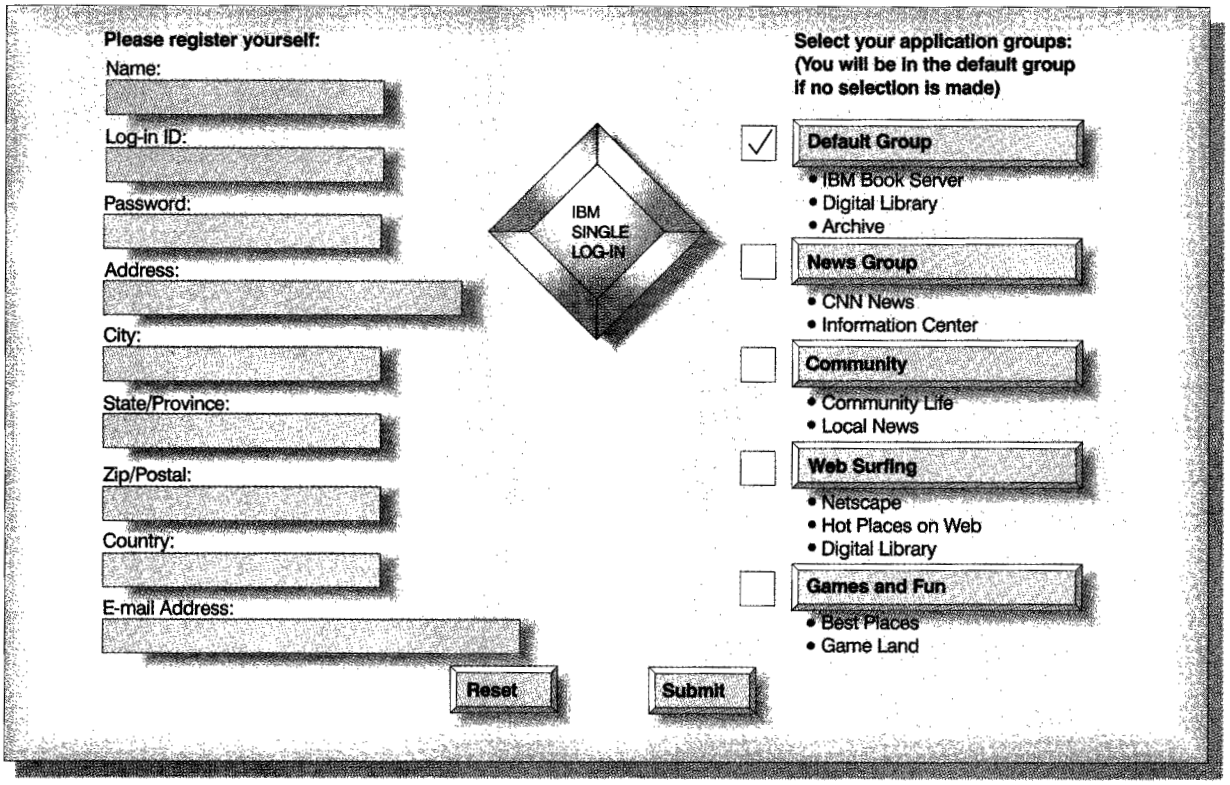
User registration subsystem

The user registration subsystem allows users to register themselves. It also provides a user with an initial list of selectable service groups. Each group contains a list of service components, and each component can reside in multiple groups. Also, a user can access multiple groups. The user registration information is stored in a persistent repository (i.e., a database) via the data manager. This registration information is propagated to the access control systems of user-selected service components. Therefore, only one registration is necessary to access all service components that are routed to WebEntree. It also enables a single user-authentication interface, i.e., a user needs only to log in once to access all of the services provided by selected service components. A sample user registration form is shown in Figure 4. The architecture and control flow for this subsystem are shown in Figure 5.

Registration mechanisms. WebEntree provides two types of user registration mechanisms: automatic and administrator assisted via electronic mail. In the automatic registration mechanism, a user fills in a registration form like the one shown in Figure 4. Then, the registration form is sent to the *registration manager* via URL after submission. The registration manager parses the user's information and calls the data manager to store the user data into the aggregator's database. Then, the registration manager invokes the registration dispatcher to register the user to the selected Web service components. The user becomes a member and gets access to the aggregated Web service immediately.

If a proofing procedure needs to be performed before a user can be registered, the administrator-assisted registration mechanism can be applied. With this mechanism, the user's registration forms are routed to a system administrator via electronic mail. If approved, the administrator sends the user's registration form to the registration manager. Then the registration manager stores the user's registration information in the aggregator's database via the data

Figure 4 A sample user registration form



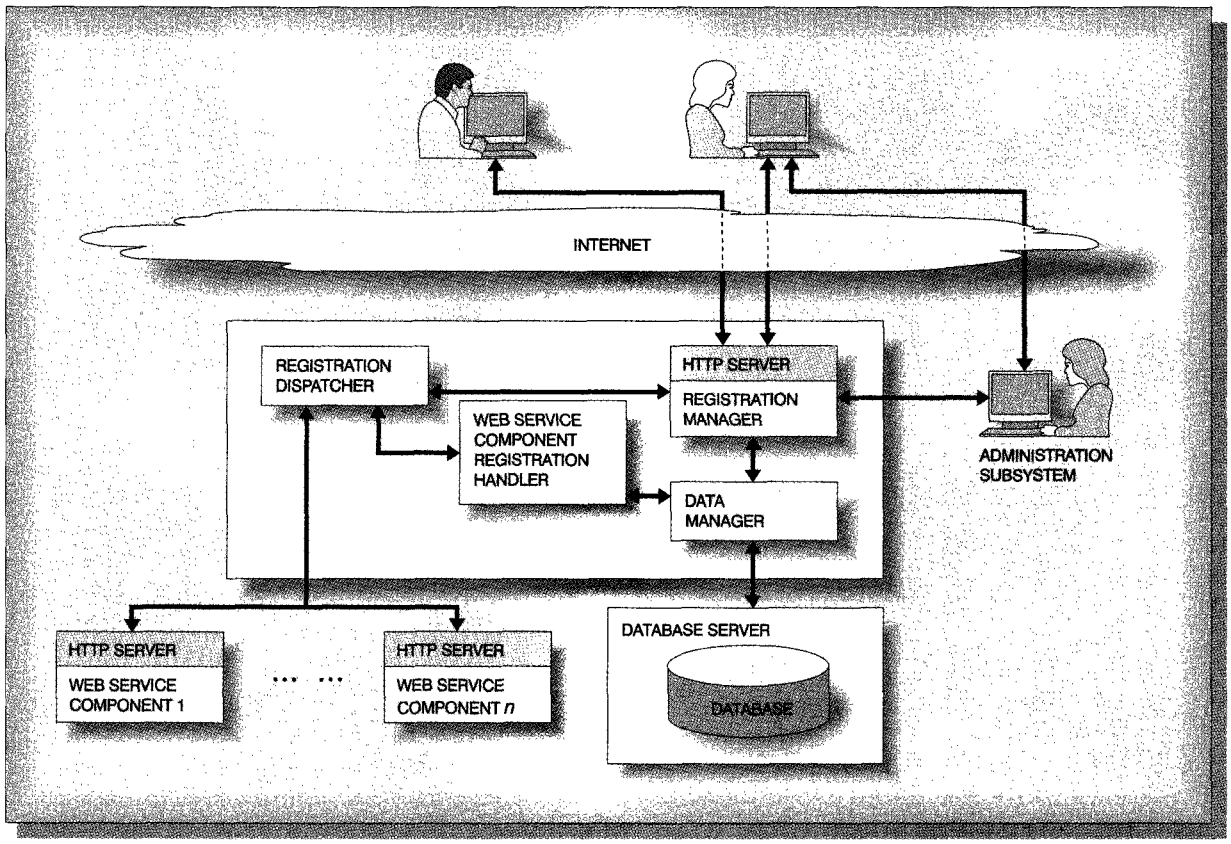
manager and invokes the registration dispatcher to propagate the registration information. The user is notified via electronic mail about membership and access information.

Registration synchronization. The registration synchronization between WebEntree and other Web service components is achieved by propagating user registration information in WebEntree to other user-selected service components via the regular registration channel provided by these components, e.g., via HTML (HyperText Markup Language) forms, client registration applets, or server registration APIs (application programming interfaces). For each Web service component, there is a special registration handler. This handler, implemented in Java code, sends registration information in the format that is acceptable by the component's registration interface. The registration method in a registration handler is customized code. Since each Web service component's registration interface can be unique, registration method automation does not seem possible in a non-affiliated environment.

The registration dispatcher is a daemon, started at aggregation service initialization time, that detects user registration and invokes the registration handlers to send registration information to the user-selected Web service components or applications.

If a user-selected Web service component provides personalized service, the user profile information, user ID, and password are sent to this component for registration. Notice that the password here is not the same password used to access the aggregator. The component password is generated by running a Java algorithm, and the algorithm produces the same password each time. To change the password is to change the algorithm, which belongs to the system and is transparent to users. The password policies can be provided from the administrator's interface, and the Java code will be dynamically generated and compiled. The generated passwords comply with the password policies specified by the Web service components. The password algorithm is implemented as a method inside the registration handler.

Figure 5 The architecture and control flow of the registration subsystem



The registration handlers are dynamically loaded at run time, an advantage of Java code. Therefore, the registration handlers can respond to registration interface and password changes for a Web service component at any time during service, as necessary. Since each password is uniquely generated at run time for each Web service component (by running the Java code), an intruder to WebEntree will not be able to find passwords to the Web service components. Each aggregated Web service component can use its own password generation methods (e.g., with encryption) or the default ones provided by this Web service aggregator.

If a Web service component does not provide any personalized service, it is unnecessary to propagate user information to it. The aggregator is registered to each Web service component with a separate ID and password. The aggregator's ID can be either dynamically generated (as for the password) or stored in the Web service component authentication table. The password generation is as described above.

This two-level registration and authentication mechanism, along with run-time password generation for accessing real content in the Web service components, has enhanced WebEntree's security, because the passwords to Web service components are not recorded anywhere. The Web service component determines whether clear password text is allowed to be transferred on the network (e.g., basic authentication) or not (e.g., digest authentication). For affiliated Web service components, one-time password or token-based authentication can be considered as well for further improvement of security, e.g., the Kerberos method.[9] SSL-based authentication can give better security, but is more expensive when considering certificate management and the involvement of a third party.[10] The overall security also relies on the security implementation for individual Web ser-

vice components, although WebEntree does not reduce their security level.

Since the same password is used by all users from WebEntree, if passwords are easily obtained from a particular Web service component (e.g., passwords are stored as clear text in the server or database), the possible damage to this component would be greater because of WebEntree's involvement. To provide better security service for each Web service component, a different password could be generated for each user. The drawback of this choice would be the additional complexity for password generation and maintenance, although the implementation would still be practical.

After registration, a user is granted a user ID, a password, and a certificate if SSL authentication is used. The user can then log in to the member service provided by the service engine of the aggregator.

A registered user can be removed by the system administrator. In addition, a user can remove himself or herself by selecting *deregistration*. During deregistration, the registration manager removes the user from the aggregator's database and invokes the registration dispatcher to remove the user from previously selected Web service components. Users can also update their service selections by themselves. Each time a user updates his or her service component group selection list, registration information will be sent to the newly selected components and deregistration information will be sent to the deselected components. Some Web service components may require more specific information than the general registration information that WebEntree provides. If so, an additional request form will prompt the user after their selection.

WebEntree service engine

The WebEntree service engine performs user authentication, service access control, gateway access to Web service components, and personalized user services. The service personalization is based on the user ID obtained during user authentication. For basic and digest authentication, the user ID, password, and authenticated server are stored in the Web browser (both Netscape Navigator** and Internet Explorer** support this feature). The user ID and password are sent to the Web server in the request header each time the user makes a URL request via the Web browser. Thus, the WebEntree service engine can provide personalized content to users based on their user ID in the request header. For other authentication mechanisms "cookies" and session-tracking techniques can be used to carry user information. The servlet APIs are useful for these operations.[11]

The architecture of WebEntree's service engine is shown in Figure 6. The Web service coordinator manages the workflow among the different components inside the service engine. For first-level authentication (users authenticate to WebEntree), the access authenticator checks the user's credential (i.e., user ID and password) based on the information in the URL request (e.g., for basic or digest authentication, user ID and password are in the request headers) and the information in the user credential object (created from the persistent data in database). If they are a match, the access authenticator sends "true" to the Web service coordinator; if not, "false" is sent. Based on the authentication result, the Web service coordinator either generates the user's personal home page or sends back an alert to inform the user of the access violation.

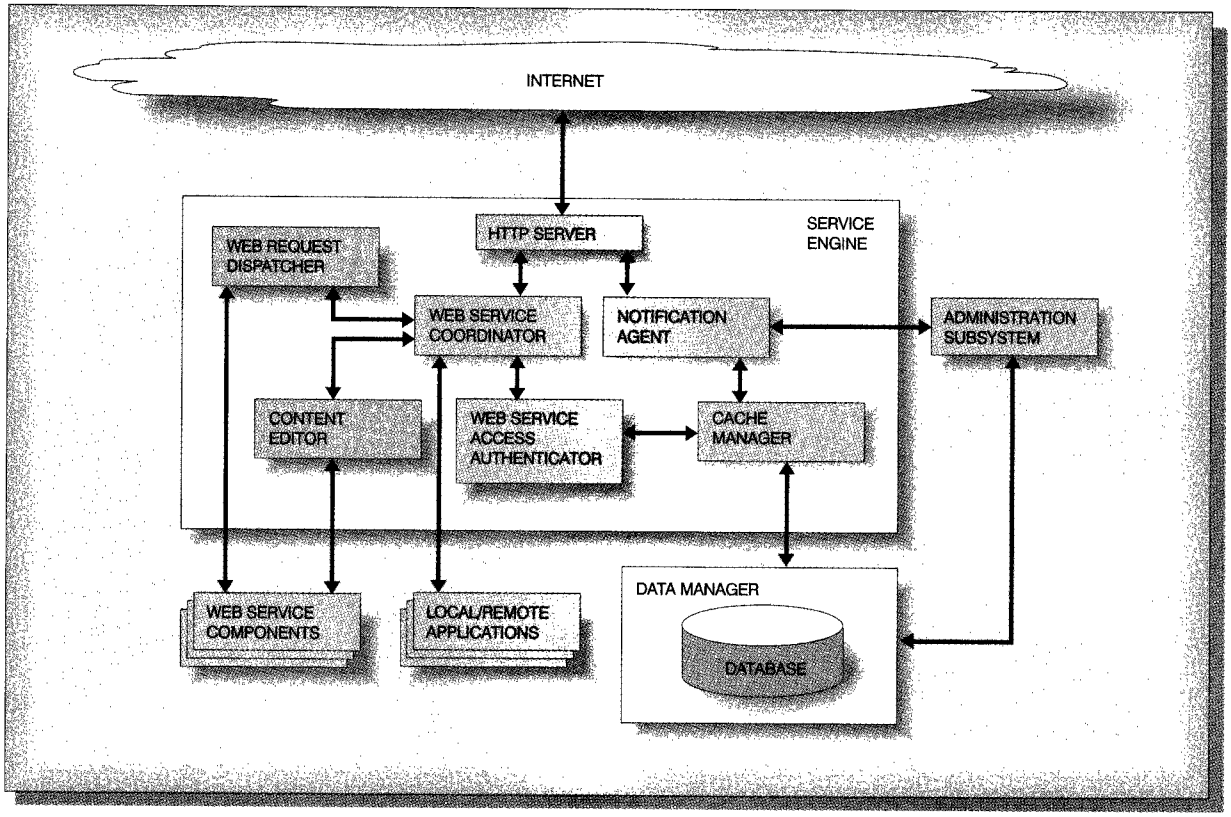
The user's personal home page is generated based on the user ID, the user-accessible component groups, and the Web service components in these groups. An access control object for each Web service component indicates by which groups it can be accessed.

After the user selects a hyperlink from his or her home page, the second-level authentication proceeds, between WebEntree and the Web service component indicated by the selected URL. The Web service coordinator sends authentication information to the component's Web server—a password for basic or digest authentication, and a certificate for SSL authentication.

After authentication, the Web service coordinator sends the user request to the Web request dispatcher. The dispatcher constructs the URL based on the parameter string passed by the user and the server map information stored in the repository. Then, the dispatcher sends the constructed URL to the corresponding Web service component, gets response content back, and sends it to the Web service coordinator.

Notice that if WebEntree and the Web service component are not on the same side of a firewall, a proxy server or a SOCKS (socket secure) server can be used for service request and access.

Figure 6 The architecture of the service engine



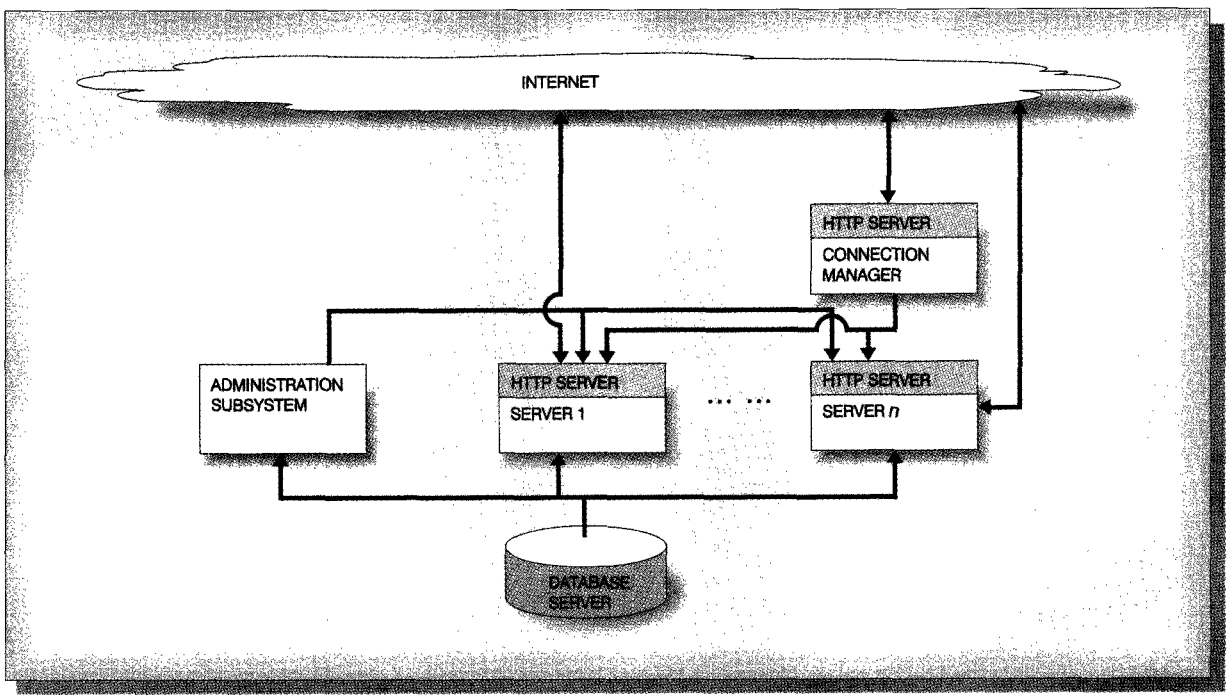
After the Web service coordinator gets the Web content from the Web service component, it sends the content to the content editor. The content editor parses the content and performs the following operations:

- Converts URLs for protocols (HTTP, FTP [File Transfer Protocol], telnet, gopher, etc.) to point to WebEntree's service engine. The URLs for images, sounds, and other multimedia data are changed to the absolute path to the original servers. It should be noticed that the "CODEBASE" attribute for Java applets should be specified for the original server.
- Adds information to the content according to the predefined service request, e.g., the WebEntree service owner's branding
- Adds specific information to the content according to user information

The local or remote applications related to content are hosted in WebEntree's Web site. The content can be in the local machine of the Web service engine or can be accessed via DCE, CORBA, etc. The Web service coordinator directly serves the content in WebEntree's Web site by invoking local and remote applications; no content parsing or transformation is necessary.

To enhance performance, a cache manager, as shown in Figure 6, manages a cache for service data. These service data are in the form of persistent data objects, such as the Web service component's information object, access control information object, and authentication information object. Each Web service component is given a unique ID when it is added to WebEntree, and the cached component information objects are retrieved by component ID. Also, the user-persistent data, such as the user profile information object and the user credential information object, can be cached for the most recent users. These user-data objects can be retrieved by user IDs. Notice that Web content is not cached here; it is handled by Web servers or cache proxy servers.[5,7]

Figure 7 The architecture for scalability and load balancing



There is a continuously running notification agent for each service engine instance (therefore for each Web server) waiting for messages from the administration subsystem. When the administration subsystem makes any changes to the persistent data, it sends a message to each notification agent. The notification agent then invokes the cache manager to update the cache from the persistent storage.

Digital certificates are involved when WebEntree or Web service components use SSL. If WebEntree uses SSL, it needs to get a server certificate. If some of the Web service components use SSL and are outside the WebEntree firewall, WebEntree needs to get client certificates for these components as well. If these components are inside the WebEntree firewall, then direct handshaking between Web clients and service components can be facilitated, and WebEntree's involvement will not be necessary.

Services provided by the service engine to registered users include changing selected service groups, changing profile information, changing passwords, and browsing over all Web service components.

For maximum performance, the service engine is scalable to multiple servers and can run in multiple boxes. The load balancing is controlled by a connection manager, as shown in Figure 7. Both the connection manager and the service engine instances are run continuously. The initial service request from a Web client browser to the aggregator goes first to the connection manager. Based on the work load in each instance of the service engine, the connection manager selects a least-loaded instance to provide the service. The first Web page is generated dynamically by the connection manager with hyperlinks to this least-loaded service engine instance.

An instance of the service engine can be added or removed by an administrator, based on the volume of user requests. This service scale adjustment operation will not interrupt the provided Web services. If an administrator makes any changes to the persistent data in the database that are relevant to the cached persistent data objects, the administration subsystem will send messages to each notification agent in each service engine instance. Each notification agent will notify the cache manager to update the cache.

Administration subsystem

The GUI-based administration subsystem provides system administrators with a convenient tool for the management of user information and Web service components, as well as for Web aggregation service customizations. It can be implemented by Java applets (based on a Web browser) or an application (based on a server). Its major functions are described in this section.

Registration processing. Administrator-assisted registration processing provides a way to retrieve user registration mail, evaluate user information and service selections, modify incoming user data, and register users to the WebEntree system.

Service management. Some basic system administration features are listed as follows:

- *User information view.* This is a view of a selected user's profile, the Web service component groups that the user has access to, and a complete list of Web service components that are accessible by the user. A user can be selected by user ID, by user name, or by group.
- *User update.* This is an interface for administrators to add or remove a user, to update a user's profile and access group information, and to change a user's password.
- *Web service component information view.* This view provides information views for a selected component. That includes the component's name, its unique ID in the system, the service it provides, the server it is located in, its invocation URL, the authentication type it uses, the groups it belongs to, etc.
- *Web service component update.* This is an interface for administrators to add or remove a selected Web service component, to update a component's information, and to assign or unassign the component to a group. A component can be assigned to multiple groups.
- *Group information view.* The Web service components are bundled into groups. For a selected group, this view shows which Web service components are inside the group, and which users have access to the components in this group.
- *Group update.* Administrators can add or remove a group, change a group's name, change a group by adding or removing Web service components, and add or remove users from a selected group.
- *Server information view.* This view shows which Web service components are in each server and which port a component is using. The server information is used for Web request dispatching and Web content transformation.
- *Server update.* A server can be added to or removed from a server control list. WebEntree service access control only applies to the servers in the server control list.
- *Service scale control.* Based on the volume of user requests, a WebEntree server can be added or removed by a system administrator.

Server administration tool for load balancing. Scalability and load balancing are provided by the service engine of WebEntree. A Web server for a service engine can be added or removed by the use of the server administration tool. The server information is stored in a configuration file in the same server as the administration subsystem. As the administration subsystem is designed to be in a single server, a file is more convenient to use than a database.

Conclusion

WebEntree, a Web service aggregator, provides an aggregated Web service by bringing other Web service systems into the service site. Although each Web service component may have its own user registration and authentication facilities, the aggregated service system performs user information and credential coordination activities and provides a single registration and log-in capability to its users. The aggregated Web service provides convenience by putting services a user wants in one place for access through a single log-in. The aggregation of Web service components from other Web sites can broaden the content, enrich the portfolio, and enhance the Web site of a Web service provider.

Currently, the Web service components are selected by the service provider who uses WebEntree. Agent technology for content discovery can be added in the future to find interesting service components. Also, user-based agent service could be added to bring free information to users, based on their interests analysis, to achieve better personalized services. This will be possible when more Web content is based on XML (eXtensible Markup Language).

The system is implemented purely in Java with Java servlets[12] running under an HTTP server, JDBC for repository or database access, and either Java applets or a Java application for the administration subsystem. New features will be added to WebEntree,

such as an advertising system, usage logs and reporting, billing and payment subsystem, and many more.

Acknowledgments

This work was supported by the Telecommunications and Media Solutions unit of IBM. I would like to thank my former colleague Khalid Asad for his initial contribution to the single log-in conceptual architecture and implementation tools. I would like to thank Denny Zhang, Chris Demas, and others for their implementation work. Also, I would like to thank Paul Grandin, Chris Demas, Andrea Macaluso, and Frank Stein for reading and commenting on this paper, and thank Frank Stein for giving me the opportunity to lead this project.

**Trademark or registered trademark of Object Management Group, Sun Microsystems, Inc., Netscape Communications Corporation, or Microsoft Corporation.

Cited references

1. J. Hursti, Single Sign-On, Department of Computer Science, Helsinki University of Technology (November 1997), available at http://www.tcm.hut.fi/Opinnot/Tik-110.501/1997/single_sign-on.html.
2. See http://www.software.ibm.com/enetwork/globalsignon/.
3. Introduction to Single Sign-On; The Open Group (1996); available at http://www.opengroup.org/security/sso/sso_intro.htm.
4. X/Open Single Sign-on Service (XSSO)—Pluggable Authentication Modules, The Open Group (1997); available at http://www.opengroup.org/onlinepubs/8329799/toc.htm.
5. R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee, "*Hypertext Transfer Protocol—HTTP/1.1*," Network Working Group RFC 2068 (January 1997); available at http://www.w3c.org/Protocols/rfc2068/rfc2068.
6. A. Dave, M. Sefika, and R. H. Campbell, "Proxies, Application Interfaces, and Distributed Systems," *Proceedings Second International Workshop in Object Orientation in Operating Systems*, Paris, France (September 1992), pp. 212–220.
7. Netscape Proxy Server Deployment Guide, Netscape Communications Corporation, available at http://home.netscape.com/comprod/server_central/eval_guide/proxy/deployment/index.html.
8. Known Web Proxy Servers Implementing PICS, World Wide Web Consortium, available at http://www.n2h2.com/pics/proxy_servers.html.
9. S. P. Miller, B. C. Neuman, J. I. Schiller, and J. H. Saltzer, "Kerberos Authentication and Authorization System," *Project Athena Technical Plan*, Section E.2.1, Massachusetts Institute of Technology, Cambridge, MA (October 1988).
10. See VeriSign, Inc. at http://www.verisign.com/.
11. JavaServer Product Documentation, Sun Microsystems, Inc.; available at http://jserv.javasoft.com/products/java-server/documentation/index.html.
12. Java Servlet Application Programming Interface White Paper, Sun Microsystems, Inc. (1996); available at http://mech.postech.ac.kr/Java/java.sun.com/products/jeeves/CurrentRelease/doc/api.html.

Accepted for publication July 13, 1998.

Yan Zhao *Adobe Systems, Inc., 345 Park Avenue, San Jose, California 95110 (electronic mail: yanzhao@aol.com).* Dr. Zhao worked for IBM from 1992 until she recently joined Adobe Systems' Advanced Technology Group as a senior computer scientist. She was an architect, a technical team leader, and a developer in several software research and development projects at IBM. Her technical interests are the Internet, Web-based content management and publishing, digital libraries, distributed computing, data modeling, and database design. She has filed several patents and provided technical consulting services for IBM marketing and external customers. Dr. Zhao received the B.S. and M.S. degrees in computer telecommunications and engineering in 1982 and 1985 from Beijing University of Posts and Telecommunications, where she was an assistant professor and lecturer until August 1988. She received the M.N.S. degree in computational mathematics in 1991 and the Ph.D. degree in computer science in 1995, both from Arizona State University.

Reprint Order No. G321-5692.