Changtao Qu, Thomas Engel, Christoph Meinel:
Implementation of a WebDAV-based Collaborative Distance Learning Environment
in Proceedings of the 28th SIGUCCS Conference on User Services (SIGUCCS 2000), ACM Press, Richmond, Virginia, USA, pp. 258-265, 10, 2000. ISBN: 1-58113-299-8.

Implementation of a WebDAV-based Collaborative Distance Learning Environment

Changtao Qu, Thomas Engel, Christoph Meinel
Institute of Telematics
Bahnhofstr. 30-32, D-54292 Trier, Germany
+49 651 975510
{qu, engel, meinel}@ti.fhg.de

ABSTRACT

In this paper, we propose a new approach for constructing Collaborative Distance Learning (CDL) environment utilizing the latest IETF specification: WebDAV. As a Web-based groupware in nature, this CDL environment enables distance learners to accomplish most of academic collaboration activities that are needed in a virtual university directly on the Web through standard Web browsers. Besides WebDAV, an enterprise-level platform: J2EE is adopted as the foundation of the CDL environment in this approach. It ensures the scalability, extensibility, and openness of the system facilitated by sets of J2EE technologies, e.g., EJB, JavaBeans, and JSP, etc. We then introduce a prototype implementation of the CDL environment called "Collaborative Desk". "Collaborative Desk" will be applied as an important way for supporting collaborative learning in a virtual university.

Keywords

Collaborative Distance Learning, Web-based Distributed Authoring and Versioning, Java 2 Platform Enterprise Edition, Enterprise JavaBeans, JavaServer Pages, Virtual University.

1. INTRODUCTION

Although the researches on computer-supported collaborative learning (CSCL) have been underway for many years, most of research work before 1990s was only concentrated on supporting collaborative learning among a small group of learners, in most cases, among on-campus students [8][12]. Before this timeline, collaborative learning was usually taken as the complement of conventional teaching methods in a traditional university, and did not take distance learners' collaboration needs into account. Since middle 1990s, with the prosperity of the World Wide Web (WWW), virtual university has begun to experience rapid growth. As a result, the research on Collaborative Distance Learning (CDL), which focuses specifically on supporting the Web-based academic collaboration amongst distance learners in a virtual university, has received more attention and gradually become the focus of CSCL [5][6][7][13].

In fact, compared with the traditional collaborative learning, the most notable change in a Web-based CDL environment is the expansion of the learner community. Instead of on-campus students, thousands of distance learners have become the main body of collaboration activities in CDL environment, and as a result, constructed a large-scale, geographically dispersed, loosely organized, and even frequently changed learner community. Such type of a learner community provides some crucial challenges to the design and construction of CDL environment.

First, it challenges our design concept.

Actually, collaborative learning has been for a long time considered only as the complement of conventional teaching methods. This is probably true in traditional universities, where the students can meet each other almost everyday and collaborative learning does not seem to be so crucial because it is relatively easy to achieve through face to face communication. But in a virtual university, collaborative learning plays actually a more important role [6], and therefore, should not be taken only as the complementary method anymore. As a matter of fact, in a distance education environment, collaborative learning is probably the most important way to inspire distance learners, who are in most cases lacking in face to face communication and collaboration opportunities compared with their on-campus comrades, to actively participate in the academic collaboration activities (e.g., engaging in collaborative problem solving), and further leads to the potential for self-inspired learning. From this perspective, CDL environment should be taken as the focal point of the virtual university design.

However, our many year's experience in the research on virtual university suggests, to date CDL environment has not occupied its deserved place in the virtual university design. In fact, we have concentrated too much on some fundamental facilities of a virtual university, e.g., digital library, courseware, etc. In most cases, we begin to construct a virtual university firstly from listing an array of courses on the Web, and then, further concentrate on the implementation of sets of essential facilities of a virtual

university, e.g., student administration system, courseware delivery system, etc. Although these essential facilities are very important for a virtual university, this design model leads usually to a negative consequence, namely, we do not give enough consideration on the CDL environment design. When the construction of CDL environment is finally put in the schedule, its design and implementation are usually limited by the existing infrastructure of a virtual university and its compatible technologies, e.g., communication protocol, computing model, developing language, etc. Because of these limitations, the power of CDL environment in supporting effective collaborative learning can usually not be fully unleashed.

In our approach, we wish to change this type of design model and explore a so-called collaboration-centric design model for the future virtual university. Put generally, throughout our prototype implementation, we always put CDL environment as the center of the virtual university design. First of all, we adopt a latest, collaboration-friendly Internet communication protocol: Web-based Distributed Authoring and Versioning (WebDAV)[9] rather than HyperText Transfer Protocol (HTTP) as the cornerstone of CDL environment in order for fully unleashing the Web's potential in supporting collaboration activities. Secondly, an enterprise-level platform: Java 2 Platform Enterprise Edition (J2EE)[18] is adopted as the foundation of the CDL environment in order to directly address the complex computing requirements put forward by CDL. In fact, J2EE is also the key to the implementation of collaboration-centric virtual university design. J2EE is a naturally open, scalable, and extensible platform taking advantage of its distributed, component-based, multi-tier architecture and some of its primary technologies [19], e.g., Enterprise JavaBeans (EJB), JavaServer Pages (JSP), Java Messaging Service (JMS), and Java Transaction Service (JTS), etc. In the next development phase, J2EE enables us to integrate our existing research work on virtual university into the prototype and further extend the prototype to a practical virtual university implementation without shaking the component-based foundation of the system. This reflects just the primary idea of the collaboration-centric virtual university design.

Second, the learner community of CDL environment challenges our choice of existing collaboration methods.

Generally, CDL can be divided into two types [2]: synchronous CDL, whose primary collaboration methods include video conferencing, real-time audio conference, Internet chat room, etc., and asynchronous CDL, whose primary collaboration methods include E-mail, newsgroups, and groupware. In fact, the characteristics of the learner community in a virtual university have determined that at least at present, asynchronous CDL should be the primary type for supporting collaborative learning on the Web. As we can imagine, the large-scale learner community often means various learners' accessibility to CDL environment. As an example, some corporate-based learners may have the access to the Internet through a high-speed local area network, while other home-based learners may only have the access through regular telephone lines with a modem. These differences in the accessibility and some other factors, e.g., organization problems across the time-zone, determine that our choice should be limited in asynchronous CDL, although in the meantime some synchronous CDL methods can be adopted as the necessary complements.

The Web-based asynchronous CDL is actually centered upon the document management and document authoring. In a typical asynchronous CDL scenario, the learners need usually to collaboratively work on a document, exchange their ideas or information (mainly by means of document exchange), and manipulate the namespace of the document repository (e.g., copy, delete, move, etc.). These document-centric collaboration activities determine that the primary applicable collaboration method in an asynchronous CDL environment is groupware. As a matter of fact, although E-mail and newsgroups are very popular and frequently used by distance learners, with the expansion of the learner community, the organization and coordination will become increasingly difficult or even impossible by using E-mail and newsgroups [5][6]. Moreover, some indispensable functionalities required by asynchronous CDL, e.g., document version control and document locking mechanism, are nearly impracticable utilizing these two simple collaboration work tools [24]. Indeed, from the inception of CDL research, groupware has been all along the researchers' focus [3][16]. Particularly, with the rapid growth of the WWW, the Web-based groupware has become the mainstream of the groupware system design since middle 1990s. Some Web-based groupware systems, e.g., BSCW (Basic Support for Cooperative Work) [4], have been already successfully applied in CDL environment [1]. In fact, with a simple client-server architecture, especially taking standard Web browser as the client program, the Web can give significant benefits in easing the development and deployment of CDL environment. However, as we will discuss in more details in the following, today's Web is not a collaboration-friendly medium in nature based on its fundamental protocol: HTTP [24]. Instead of HTTP, our final choice is locked on a latest, collaboration-friendly Internet communication protocol: WebDAV. Indeed, WebDAV can directly address the collaboration needs provided by the large-scale learner community of CDL environment, and therefore, is more appropriate to be adopted as the new cornerstone of CDL environment.

2. WEBDAV: NEW CORNERSTONE OF CDL ENVIRONMENT

HTTP has served as the cornerstone protocol for the WWW since 1990. However, HTTP is actually not a collaboration-friendly protocol in nature [24]. HTTP works well for static documents intended for viewing, but does not provide clients with rich authoring capabilities to handle these documents, e.g., document version control, document locking mechanism, etc. Several obvious shortcomings of HTTP concerning the Web-based authoring, e.g., drawbacks in support for parameter marshaling, support for document operations, support for operating on hierarchies of document, and error reporting [22], have constructed obstacles to the Web-based CDL. In fact, WebDAV is just the protocol that is designed to eliminate these obstacles.

WebDAV protocol is worked out by WebDAV working group of Internet Engineering Task Force (IETF) and was

partly finalized in the form of IETF RFC (Request for Comments) in February 1999 [9]. Because HTTP has already proven itself as a flexible, universal protocol for transferring data despite of several shortcomings concerning Web-based collaboration, WebDAV intends to take a "shortcut" and provides the downward compatibility with HTTP. It extends HTTP/1.1 and provides a coherent set of new methods, headers, eXtensible Markup Language (XML)-based request and response entity body formats to directly support collaborative work on the Web.

In general, WebDAV provides following four basic capabilities [9]:

Properties: The ability to create, remove, and query meta-information about documents and the ability to specify properties that provide links connecting media types that are unable to contain embedded links.

Collections: The ability to create sets of related documents and to retrieve a hierarchical listing of their members.

Locking: The ability to control access to resources to avoid "lost updates" in a distributed, multi-user-authoring environment. Instead of strictly exclusive locking mechanism that locks the entire document for each user, WebDAV allows the shared locking of a document.

Namespace Manipulation: The ability to copy, move, or delete Web resources and collections of resources.

In addition, WebDAV has also several other functionalities which are now still under exploration by IETF. Relatively speaking, these additional capabilities are more important for CDL, although the four basic functionalities of WebDAV are already important enough. The additional capabilities are [24] :

Advanced Collections: The ability to extend the basic WebDAV collections to cater for advanced features, such as references and ordering. References, which act like symbolic links or shortcuts, can allow resources to be shared. On the other hand, ordering brings sorting facilities, allowing better manipulation of Web resources.

Versioning and Configuration Management: The ability to support operations such as check-out, check-in, and retrieval of the history list. Built on top of the versioning layer is the configuration management layer, which provides support for workspaces and configurations, allowing versioned collections of versioned resources to be worked on.

DAV Searching and Locating (DASL): The ability to efficiently search for resources based upon a set of client supplied criteria. The actual query grammar is not specified by DASL and the server may provide a number of different query grammars such as SQLX. The query grammar may support searching the properties of resources and/or their content.

Access Control: The ability to set and clear ACLs (Access Control Lists). This functionality is crucial for allowing cooperators to remotely add and remove people from the list of cooperators on a single resource. At its most general, this activity becomes access control not just for DAV, but for the entire Web.

With regard to CDL, WebDAV can provide following four sides of support:

First, the support for deployment.

With the expansion of the learner community, the client-side deployment of CDL environment always means great challenge. Generally, because learners may have various machines, various operating systems, various network environments, and various Web browsers, the client-side deployment, e.g., installing some proprietary softwares or installing plug-ins for Web browsers, implies in most cases a complex, cumbersome, and problem-prone process. In fact, WebDAV's support for the deployment of Web-based CDL environment is just one of its most notable features. As a protocol-level specification, WebDAV enables the learners to "in-place" (means directly on the remote server) manipulate Web resources through standard Web browsers. That is to say, no client-side deployment is needed in a WebDAV-based CDL environment, a standard Web browser is already enough for the learners to accomplish all the collaboration activities wherever they are in the world, just like viewing Web pages through these browsers.

Second, the support for collaborative authoring.

Collaborative authoring is the major collaboration activity in a CDL environment. Exactly, the support for collaborative authoring is just the primary design goal of WebDAV. In a distributed, document-centric CDL environment, documents need usually to be generated, presented, updated, accessed, shared, and exchanged directly on the Web. WebDAV defines a set of methods to support "in-place" document editing, document locking, and document version control. These methods provide all functionalities that are necessitated by collaborative authoring in a CDL environment.

Third, the support for security management.

Although the security issue is not so crucial in a CDL environment as in a business-to-business or business-to-consumer electronic commerce application, the security is still an important aspect to ensure collaboration activities against malicious or non-malicious interference. WebDAV's support for security management in a CDL environment focuses primarily on the secure access to Web resources utilizing its definition of access control. The access control enables the learners to control other learners' access to the resources. Moreover, because the collaboration needs are frequently changing in a CDL environment, WebDAV has also defined methods to enable the learners to easily adjust ACLs.

Finally, the support for information exchange.

Information exchange may occur at every phase of collaboration activities. WebDAV supports information exchange primarily taking advantage of its capability of manipulating properties (meta-data) of Web resources (e.g., create, modify, or delete properties). In a typical information exchange scenario, the learners can add description

information to their own documents, or make some comments on other learners' documents by setting properties. Likewise, the instructors can make advice on the group work (often represented as a collection) by setting collection's property. WebDAV's property definition is completely based on XML in a format known as the Dublin Core [21], which provides the learners with a quite flexible way to accomplish information exchange. Moreover, the property manipulation can be also combined with WebDAV's DASL functionality in order for locating Web resources. In a CDL environment, effective searching is also an important side of support for information exchange.

In fact, besides all sides of support listed above, the most notable feature that a CDL environment can achieve taking advantage of WebDAV is strong interoperability. Although most of WebDAV's functionalities can be also implemented in a HTTP-based CDL environment utilizing some proprietary methods, these methods can not be recognized by other client-side or server-side applications, since they all belong to the proprietary extensions to HTTP. In contrast, WebDAV's extension to HTTP is a pure standard in the real sense. Just like HTTP methods: "get" or "post", all WebDAV methods (e.g., propfind, lock, copy, move, etc.) can be recognized by other WebDAV-compliant applications. We can expect, with the large range of acceptance of WebDAV protocol, the WebDAV-based CDL environment may achieve stronger interoperability with other WebDAV-compliant clients or other WebDAV-compliant CDL environment. As an example, today we can already "in-place" edit a document using WebDAV-compliant Microsoft Office 2000, or "in-place" manipulate the document repository through drag and drop using WebDAV-compliant Internet Explorer 5.0 in our prototype: Collaborative Desk (C-Desk). Besides these popular client-side applications, WebDAV has also found many server-side, enterprise-level application cases which can fall into the category of groupware [23]. These applications can to a certain degree presage the prosperity of the WebDAV-based CDL environment in the near future.

3. J2EE: ENTERPRISE-LEVEL PLATFORM FOR CDL ENVIRONMENT

In our approach, J2EE is adopted as the fundamental platform for addressing some crucial computing requirements put forward by large-scale learner community of CDL environment (e.g., availability, scalability, extensibility, and security). J2EE is actually a platform specification defined by Sun Microsystems company to enable enterprise-level solutions for developing, deploying, and managing multi-tier, server-centric applications [18]. J2EE provides a server-centric, component-based, e-business-oriented, and multi-tier application architecture that can support Web-based, enterprise-level applications facilitated by sets of latest Java technologies [19], e.g., EJB, JSP, JMS, JTS, etc. Generally, we take J2EE as the fundamental platform for the CDL environment mainly out of following considerations:

First, J2EE provides support for extremely robust computing in a CDL environment.

The large-scale learner community provides more crucial requirements on CDL environment. Among these requirements, two notable aspects are high availability (the capability of providing 24-hour, no-downtime services) and scalability (the capability of supporting large number of concurrent learners). Taking advantage of a server-centric architecture, J2EE can provide support for these two aspects through various ways [19]. For example, some services can be partitioned across multiple servers so as to achieve workload balancing, some application components can be replicated so as to support instant failover and recovery, and a single logical database can be split across multiple physical databases so as to increase reliability, etc. Relatively speaking, these technical methods, which are crucial for a distributed CDL environment, are difficult, or even impossible, to realize based on a "simple" platform (e.g., sole Web server).

Second, J2EE provides an open architecture that enables us to extend the CDL environment in order to meet some new needs put forward by a virtual university.

In the near future, we plan to extend C-Desk to a practical virtual university implementation. With this purpose, not only some other collaboration methods (e.g., video conferencing) but also some other function modules of virtual university (e.g., courseware system) will be integrated into the system. In this process, we can get actually three-level support from J2EE platform. In the product-level, we can directly adopt some ready-made third-party products; In the component-level, we can utilize some ready-made EJBs or JavaBeans developed by third-party producers; In the protocol-level, we can integrate some latest Internet or non-Internet protocols into the system. In fact, these three types of integration have all found corresponding application cases in a J2EE platform [20]. In the next development phase, we plan to integrate WAP (Wireless Application Protocol) into C-Desk in order to support the access through mobil phone. This sort of integration will further address the potential needs provided by a virtual university.

Third, J2EE can simplify the developing process of the CDL environment.

As a matter of fact, one of the most important features of J2EE is its capability of automatically managing some system-level services [19]. These services, which are usually called middleware services, include security management, transaction management, component life-cycle management, and persistence management, etc. J2EE can shield these services and their underlying technologies from developers, which enables the rapid development of CDL environment. Moreover, as we will describe in the following, the computing model of J2EE is mainly based on some reusable components such as EJB, JavaBeans, and JSP. These reusable components can also simplify the developing process.

Finally, J2EE provides us with the opportunity to realize some currently "unavailable" functionalities of WebDAV.

WebDAV is at present not yet completely finalized. The most important part of WebDAV: access control is still

under developing by IETF. By adopting J2EE, we can temporarily utilize the standard security services provided by J2EE (including authentication, authorization, and ACLs) to realize security management in C-Desk. After WebDAV is completely finalized in the near future, we can integrate WebDAV's access control into C-Desk. Actually, the similar approach is currently also applicable to the version control functionality of WebDAV.

Since J2EE is a relatively new specification, there are currently not too many J2EE-compliant products in the market. At present, the available products include iPlanet Application Server 6.0, BEA WebLogic Server 5.0, Inprise Application Server 4.0, and IBM WebSphere Application Server 3.0. In C-Desk, we adopt IBM WebSphere Application Server Advanced Edition 3.02 (WAS AE) to play the role of J2EE platform. It constructs the foundation of the whole system.

4. C-DESK: A PROTOTYPE OF CDL ENVIRONMENT

As its name implies, C-Desk is mainly used to support academic collaboration activities among distance learners in a virtual university. These collaboration activities can be described through a typical collaboration scenario in C-Desk.

In general, all collaboration activities in C-Desk are organized into different projects. A project, which can be further organized in a hierarchy, contains all the data and processes needed to perform group tasks. As the beginning, the instructor constructs a collaboration project on the server (creating a collection and defining its ACLs) and invites the learners to participate in it (utilizing "bulletin board" in C-Desk, or utilizing E-mail). In the project, the learners can be organized into a few groups (sub-collections), either spontaneously or according to some rule. Similarly, ACLs should also be defined upon these sub-collections.

When the collaboration process begins, the learners can upload their original ideas (documents) or presentations onto the remote server, and at the same time, define the ACLs of these documents. Based on some common ideas, some shared working reports can be created. When the learners wish to modify these shared documents, they must firstly lock them, and then, either "in-place" edit them utilizing some WebDAV-compliant softwares (e.g., Microsoft Office 2000), or download them for the local editing. When the learners wish to exchange information, they can "set" or "get" the properties of collections or documents. Additionally, according to different authority, the learners can manipulate the namespace of the project, either upon collections, or upon documents. Moreover, some auxiliary functions (e.g., searching, notifying through E-mail or "bulletin board") can be used in this process in order to further achieve effective collaboration.

4.1 System Architecture

In figure 1 we illustrate the system architecture of C-Desk.

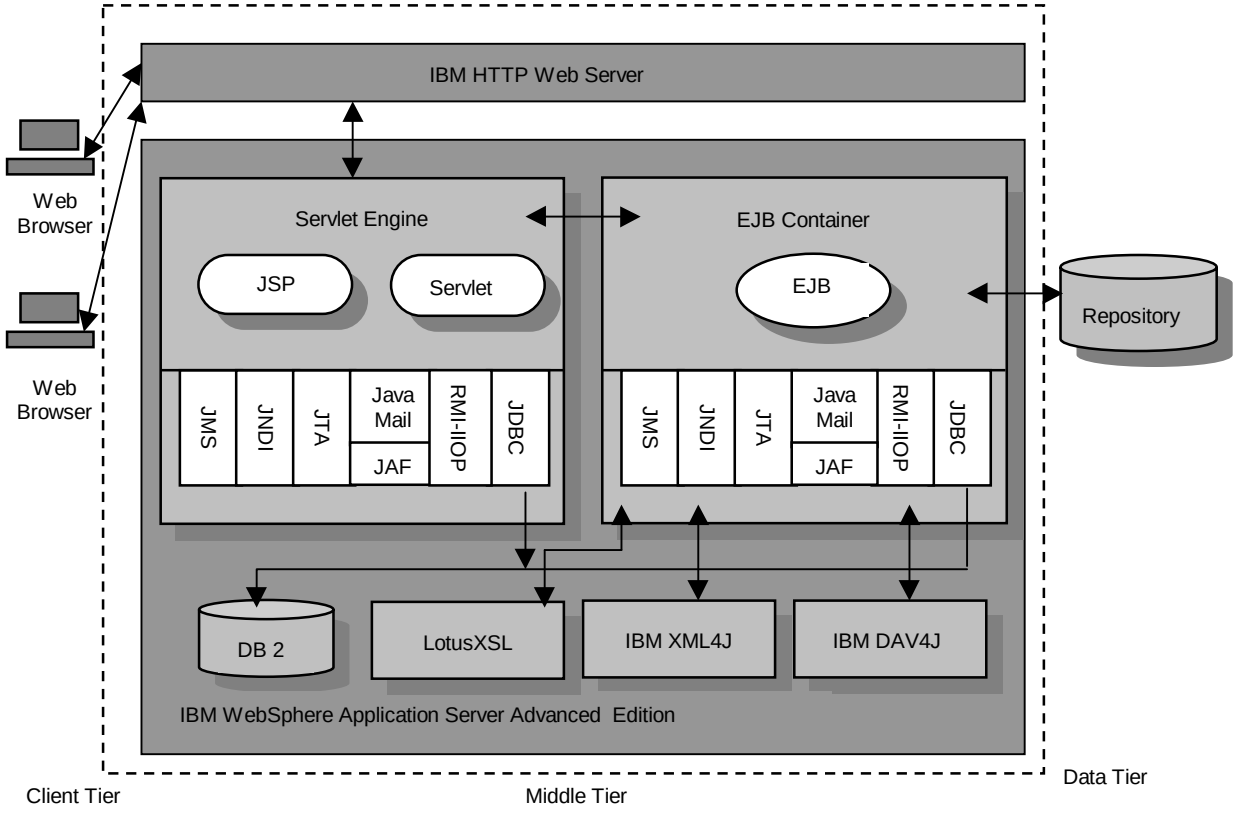

Figure 1. System architecture of C-Desk

C-Desk is constructed on Windows NT 4.0, Server Edition. The standard Web server is IBM HTTP Server 1.3.6.2. It is an Apache Web server extended by IBM by means of adding some additional extension modules in order to improve its performance, security, and usability [10].

The backbone of C-Desk is WAS AE. It is a J2EE-compliant product with an EJB container that can support EJB specification 1.0, and a servlet engine that can support both JSP specification 1.0 and Java Servlet specification 2.1 [15]. Both servlet engine and EJB container can provide support for sets of standard APIs defined by J2EE, e.g., JNDI, JDBC, JMS, etc.

The WebDAV functionalities in C-Desk are realized by IBM DAV4J 1.0.34 [11]. DAV4J consists of a client-side API and a server-side servlet to provide full support for class-2 WebDAV implementation (implying properties, collection, namespace, and locking). Because the WebDAV methods, headers, request and response entity bodies, and status codes are all directly based on XML, an XML parser: IBM XML4J is used in the system to handle XML semantics. In addition, an XSL (eXtensible Stylesheet Language) processor: LotusXSL is applied to transfer some returned XML-based results (e.g., properties of a document) into HTML codes. XML4J and LotusXSL are built-in modules of WAS AE. In contrast, DAV4J is an extension module which is added by us for implementing WebDAV functionalities. WAS AE has also another built-in module: IBM DB2 Universal Database 6.1. It is used to store the persistent data of EJBs.

C-Desk is designed utilizing a server-centric, three-tier architecture. The middle tier is constructed by WAS AE. The data tier is the document repository. It can be located on the same server with WAS AE, or located on another server. The client tier is standard Web browsers. Generally, they need not any sort of plug-in or client-side installation.

4.2 WebDAV Implementation: IBM DAV4J

At present, there are two popular methods to implement WebDAV functionalities. The first method is using Apache mod_dav module. It is an extension module of Apache Web server written in C language and can realize class-2 WebDAV implementation [22]. The other method is using IBM DAV4J. It accomplishes class-2 WebDAV implementation utilizing Java language. In table 1 we describe a comparison between IBM DAV4J and Apache mod_dav.

Table 1. A comparison between IBM DAV4J and Apache mod_dav

	IBM DAV4J	Apache mod_dav
Platform support	Platform-independent	Win32, Unix
Implementation	Web server with Java servlet support + XML parser (prefer to IBM WebSphere Appl. Server)	Apache Web server+Expat XML parser
WebDAV support	Class 2	Class 2
Multiple backend repository support	Yes	Yes
Client/Server communication	HTTP, RMI, IIOP	HTTP
HTTP-protocol-level developing support	Yes	Yes
WebDAV-protocol-level developing support	Yes	No
Object-Oriented-API-level developing support	Yes	No
Source code	Java, promise of open source in the near future	C, open source

Compared with Apache mod_dav module, DAV4J provides a more flexible means to implement WebDAV functionalities. It takes advantage of servlets and the Java object model to provide a rich, object-oriented implementation of WebDAV that can be easily updated as WebDAV specification is extended. In addition, we choose DAV4J to implement WebDAV functionalities also out of following considerations:

First, on the client side, the DAV4J client API provides a number of Java classes and interfaces that abstract the semantics of resources on the Web which are controlled by WebDAV functions. This abstraction simplifies client application development by relieving programmers from the details of low-level HTTP protocol conventions, e.g., getting and setting headers, creating and parsing entity bodies, examining response codes, etc. At the same time, the API provides a framework for protocol independent communication between client and server applications. Because all access is abstracted in Java interfaces and classes, it is possible to distribute method access through client proxy stubs and server skeleton classes for communication through HTTP, RMI (Remote Method Invocation), and IIOP (Internet InterORB Protocol). This feature of DAV4J client API is very important for the WebDAV implementation in a system with distributed object architecture, e.g., C-Desk.

Second, on the server side, DAV4J is designed to support access to multiple backend repository managers fully based on WebDAV specification. These repository managers can control

the resources and are often optimized to manage resources of a particular type. DAV4J factors out all the repository specific behavior into simple subsystem interfaces that can be implemented on top of any number of repository managers. This allows repository managers to provide a WebDAV interface without knowing hardly anything about WebDAV semantics or communication protocols.

Finally, compared with Apache mod_dav which is implemented utilizing C language, Java-based DAV4J is more compatible with J2EE platform. We can implement some currently "unavailable" WebDAV functionalities utilizing the standard APIs and standard services provided by J2EE platform, and at the same time, achieve a seamless integration with DAV4J.

In C-Desk implementation, we adopt stateless session EJB to directly invoke DAV4J API to implement corresponding business logic (implying repository manipulation), and use JSP components to implement interaction with the user's browser. JSPs are also applied to control the server-side execution logic. This design style complies also with the popular Model-View-Controller (MVC) design pattern. The view is Web browser, the model is EJB, and the controller is JSP. Because JSPs can directly include JavaBeans and in-line Java codes, it is easy to realize corresponding execution logic based on them.

4.3 Computing Model: JSP, JavaBeans and EJB

In figure 2 we illustrate the data flow of a WebDAV implementation scenario: create a collection on the remote server. It describes the typical computing model in C-Desk, which is mainly based on JSP, JavaBeans, and EJB components.

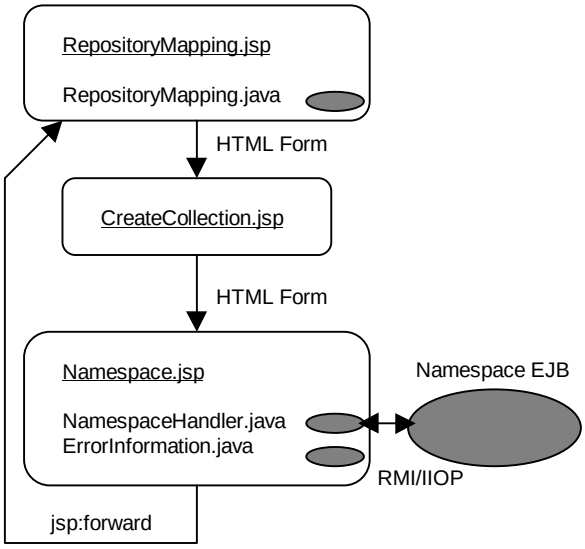

Figure 2. Data flow of a WebDAV implementation scenario

In figure 2, RepositoryMapping.jsp is a JSP component which is directly presented to the user's Web browser with dynamically generated repository mapping. It includes a JavaBean: RepositoryMapping.java that is used to get the sub-collections and documents under the current collection. The sub-collections are further embedded into the RepositoryMapping.jsp as HTML links. The users can navigate into the sub-collection by clicking corresponding link, which recursively invokes the JavaBean: RepositoryMapping.java to get its sub-members.

When the user clicks "create collection" button (HTML form), another JSP component: CreateCollection.jsp is invoked to gather the user's input information (in this scenario, the name of the new collection). After that, a key JSP component in the data flow: Namespace.jsp is invoked through HTML form

The Namespace.jsp is responsible for handling all the requests concerning namespace manipulation on the document repository. It includes two JavaBeans: NamespaceHandler.java is used to interact with Namespace EJB for implementing the user's requests (in this scenario, create a new collection on the remote server), and ErrorInformation.java is used to catch exceptions and display error message to the users.

Actually, all actions concerning namespace manipulation are implemented by a stateless session EJB: Namespace EJB. It consists of a home interface that defines methods used to create instance of EJB, a remote interface that is used by the client to indirectly invoke the business methods defined in the EJB, and a bean class that encapsulates all the concrete, user-implemented (by using DAV4J API) business methods. In the process of interaction with EJB, the JavaBean: NamespaceHandler.java does not work directly with the bean class of the Namespace EJB. In fact, when EJBs are deployed into the container, the EJB container automatically generates corresponding classes of the remote interface and home interface. In addition, the container is also responsible for generating some stub and skeleton classes which are used for remote method invocation.

So, from the perspective of the client, the EJB is only represented by two interfaces. Most of the complex handling in the interaction process, e.g., generating class instances, managing actions of the EJB container, etc., are shielded from developers. In JavaBean: NamespaceHandler.java, we need only look up the EJB home class utilizing JNDI service provided by WAS AE, and then, directly invoke the business methods via EJB's remote interface. The whole interaction is based on RMI over IIOP, which enables the clients (in this scenario, JavaBean: NamespaceHandler.java) and the EJBs (in this scenario, Namespace EJB) to exist in different runtimes on different systems on a network. In fact, the support for distributed computing is just the primary advantage of EJB. It is also the reason why we decide to adopt EJB-centric computing model to implement a distributed, extensible CDL environment.

As the last step of the data flow, the RepositoryMapping.jsp is invoked again by Namespace.jsp through jsp:forward action to update the presentation in the user's browser.

4.4 Security Management

In C-Desk, security is handled almost entirely by WAS AE and we need not write any code to implement security logic. In general, WAS AE security sits on top of the operating system security and the security features provided by the Java language

[14]. In turn, the operating system can also provide support services to the WAS AE security. In C-Desk, for example, we directly configure WAS AE to check security principals (users, groups) against the user registry of Windows NT. In addition, WAS AE can also automatically utilize the security information provided by EJB during delegation.

Utilizing the security management wizard provided by WAS AE administrative console, we have easily accomplished the security configuration of C-Desk. WAS AE supports four types of authentication methods: basic, digest, custom, and certificate [14]. At the current development phase, we adopt basic authentication based on Windows NT user registry.

With regard to authorization, WAS AE provides both application-level and method-group-level authorization policy to protect Web resources [14]. The application-level policy can be applied to the whole application, while the method-group-level policy is applied to the methods of EJB and JSP components. WAS AE predefines six method groups [14], e.g., ReadMethods (implying GET methods of beans and GET and POST methods of Web resources), WriteMethods (implying SET methods of beans and PUT methods of Web resources), etc. These method groups can cover most of the repository manipulation methods required by C-Desk. In addition, we can also define some customized method groups to realize some special authorization purpose.

5. CONCLUSIONS

Our approach for constructing CDL environment based on J2EE platform and WebDAV protocol has made full use of the existing technologies which can be applied to support Web-based collaboration to date. On the one hand, the collaboration-friendly WebDAV protocol enables us to fully unleash the Web's potential in supporting collaboration activities; On the other hand, the high-performance J2EE platform enables us to avoid complex, trivial, and in most cases repeated low-level development work. Moreover, our experience with the prototype also suggests, a high-performance, ready-made platform (e.g., WAS AE) can not only simplify the developing process, but also can ensure the strong extensibility and robustness of the system.

In our opinion, the application prospects of WebDAV-based CDL environment depend to a great extent on the maturity of WebDAV. Particularly, we place special expectation on its specifications of DASL, version control, and platform-independent ACLs. These more advanced functionalities of WebDAV will determine the practicality of the WebDAV-based CDL environment.

6. REFERENCES

[1] Agostinho, S., G. Lefoe, and J. Hedberg, "Online Collaboration for Learning: A Case Study of a Post Graduate University Course," in *Proc. of the Third Australian World Wide Web Conference (AusWeb97),* Southern Cross University, Australia, July 1997.

[2] Aoki, K., and D. Pogroszewski, "Virtual University Reference Model: A Guide to Delivering Education and Support Services to the Distance Learner," *Online Journal of Distance Learning Administration*, 1(3), Fall 1998, Available at: http://www.westga.edu/~distance/aoki13.html

[3] Becker, D., and M. Dwyer, "The Impact of Student Verbal/Visual Learning Style Preference on Implementing Groupware in the Classroom," *Journal of Asynchronous Learning Networks,* 2(2),1998.

[4] Bentley, R., W. Appelt, U. Busbach, E. Hinrichs, D. Kerr, S. Sikkel, J. Trevor, and G. Woetzel, "Basic Support for Cooperative Work on the World Wide Web," in *International Journal of Human-Computer Studies: Special issue on Innovative Applications of the World Wide Web,* 827-846, 1997.

[5] Bremer, C., "Design of a Group Oriented, Virtual Learning Environment," in *Proc. of BITE*, Maastricht, Holland, March 1998.

[6] English, S., and M. Yazdani, "Computer Supported Cooperative Learning in a Virtual University," Available at: http://www.media.uwe.ac.uk/masoud/author/jcal.htm

[7] Fowler, T., J. B. Gasen, L. Roberts, and S. Saltzberg, "Collaborative Learning Using Technology: Issues and Approaches," in *Proc. of CAUSE Annual Conference*, 1996.

[8] Gasen, J. B., and J. J. Preece, "Collaborative team projects: Key issues for effective learning," *Journal of educational technology systems,* 24(4), 181-194, 1996.

[9] Goland, Y. Y., E. J. Whitehead, A. Faizi, S. Carter, and D. Jensen, "HTTP Extensions for Distributed Authoring-WEBDAV," RFC 2518, February 1999.

[10] IBM Corp., "IBM HTTP Server," Available at: http://www-4.ibm.com/software/webservers/httpservers/

[11] IBM Corp., "IBM WebSphere DAV4J," Available at: http://www.alphaworks.ibm.com/tech/DAV4J

[12] Kumar, V. S., "Computer-Supported Collaborative Learning: Issues for Research," Available at: http://www.cs.usask.ca/grads/vsk719/academic/890/project2/project2.html

[13] Miller, M. D., and T. C. Padgett, "Redesigning the Learning Environment for Distance Education: An Integrative Model of Technologically Supported Learning Environments," *Online Journal of Distance Learning Administration*, 1(1), Spring 1998, Available at: http://www.westga.edu/~distance/miller11.html

[14] Nagaratnam, N., "IBM WebSphere Standard/ Advanced 3.02 Security Overview," Available at: http:// www-4.ibm.com /software /webservers / appserv/security.pdf

[15] Reser, J., "IBM WebSphere Application Server Advanced Edition architecture," Available at: http://www-4.ibm.com/software/ developer/ library/ wsarchitecture /wsarchitecture.html

[16] Schrum, L., and T. Lamb, "Groupware for Collaborative Learning: A research Perspective on Processes,

Opportunities, and Obstacles," *Journal of Universal Computer Science,* 2(10), 1996.

[17] Stein, G., "mod_dav: a DAV module for Apache," Available at: http://www.webdav.org/mod_dav/

[18] Sun Microsystems Corp., "Java 2 Platform Enterprise Edition Specification," version 1.2, Available at: http://java.sun.com/j2ee/ download.html

[19] Thomas, A., "Java 2 Platform, Enterprise Edition: Ensuring Consistency, Portability, and Interoperability," Available at: http://www.javasoft.com/j2ee/j2eeAnneThomas.pdf

[20] Violleau, T., and U. Shetty, "eMobile End-to-End Application Using the Java 2 Platform, Enterprise Edition," Available at: http://developer.java.sun.com/developer/ technicalArticles/whitepapers/javaone00/eMobileApplet.pdf

[21] Weibel, S., J. Kunze, C. Lagoze, and M. Wolf, "Dublin Core Metadata for Resource Discovery," RFC 2413, September 1998.

[22] Whitehead, E. J., "Lessons from WebDAV for the Next Generation Web Infrastructure," Available at: http://www. ics. uci. Edu / ~ejw / http – future / whitehead / http_pos_ paper.html

[23] Whitehead, E. J., "WebDAV State of Adoption Report," January 2000, Available at: http:// www.webdav.org/

[24] Whitehead, E. J., and Y.Y. Goland, "WebDAV: A network protocol for remote collaborative authoring on the Web," in *Proc. of the Sixth European Conference on Computer Supported Cooperative Work (ECSCW'99),* Copenhagen, Denmark, 291-310, Sept. 1999.