

# A Practical Software Architecture for Virtual Universities<sup>1</sup>

Peifeng Xiang, Tsinghua University, China

Yuanchun Shi, Tsinghua University, China

Weijun Qin, Tsinghua University, China

---

## ABSTRACT

*This article introduces a practical software architecture called CUBES, which focuses on system integration and evolvement for online virtual universities. The key of CUBES is a supporting platform that helps to integrate and evolve heterogeneous educational applications developed by different organizations. Both standardized educational applications that follow popular e-learning standards, such as SCORM and CELTS, and unstandardized ones can be integrated into CUBES with ease. A regularly used virtual university based on CUBES that includes educational applications covering the entire spectrum of the learning and management process is introduced to examine the usability of CUBES. The authors hope that new architectures, tools, and methods are explored to reduce the effort in building and evolving virtual universities.*

*Keywords: data integration; distance education; software architecture; virtual university; Web-based applications*

---

## INTRODUCTION

### Motivation

During the last decade, numerous Web-based learning systems, such as Blackboard (2004) and WebCT (2004) have been developed to provide learning environments for virtual universities. Most of these systems intend to implement a series of software modules by themselves to

support the entire online learning and management process. However, many practical systems, especially those used in universities for distance learning, are developed in a collaborative manner. Some modules come from corporations or other organizations; others have to be developed by universities to meet special requirements. In addition, these modules must evolve constantly.

Therefore, how to integrate these heterogeneous software modules into a seamless learning environment becomes a big challenge. One possible solution is the standardization of the system architecture. In recent years, many efforts have been put into this area; for example, IEEE's LTSA (IEEE LTSC, 2001), IMS's ongoing Abstract Framework (IMS, 2003), CMU's service-oriented LSA (CMU, 2001), MIT's OKI (OKI, 2004), and SCORM (ADL, 2001). These standardized architectures reduce the efforts of integration and evolution but are still insufficient due to the following reasons:

- To define a widely accepted and practical architecture proves a hard work. IEEE's LTSA, although widely accepted, is a high-level conceptual architecture and insufficient in practical implementation. Some others, such as MIT's OKI, IMS's Abstract Framework, and CMU's LSA, intend to define the architecture in more details. But they are not widely accepted and differ from each other in many aspects. SCORM is probably the most popular e-learning standard nowadays but only covers part of the system architecture.
- To thoroughly define each module in the system architecture is even more difficult. For example, the goal of IMS's Abstract Framework is to define the required applications, services, and components in the learning environment. However, it seems a long way before accomplishment and much longer before it is widely accepted.
- Standards hardly can cover newly developed modules, such as real-time interactive virtual classrooms and adaptive learning tools.
- Some legacy modules conflicting with standards have to be integrated. It may

not be a short time before standardized modules replace all of them.

It is clear that integration and evolution must be based on popular e-learning standards, but only these standards are not enough. In a practical virtual university, efforts have to be made to address the problems that standards do not care.

### **A New Software Architecture**

CUBES is a new software architecture for virtual universities. It is the result of years of working in a practical virtual university project (Tsinghua, 2003).

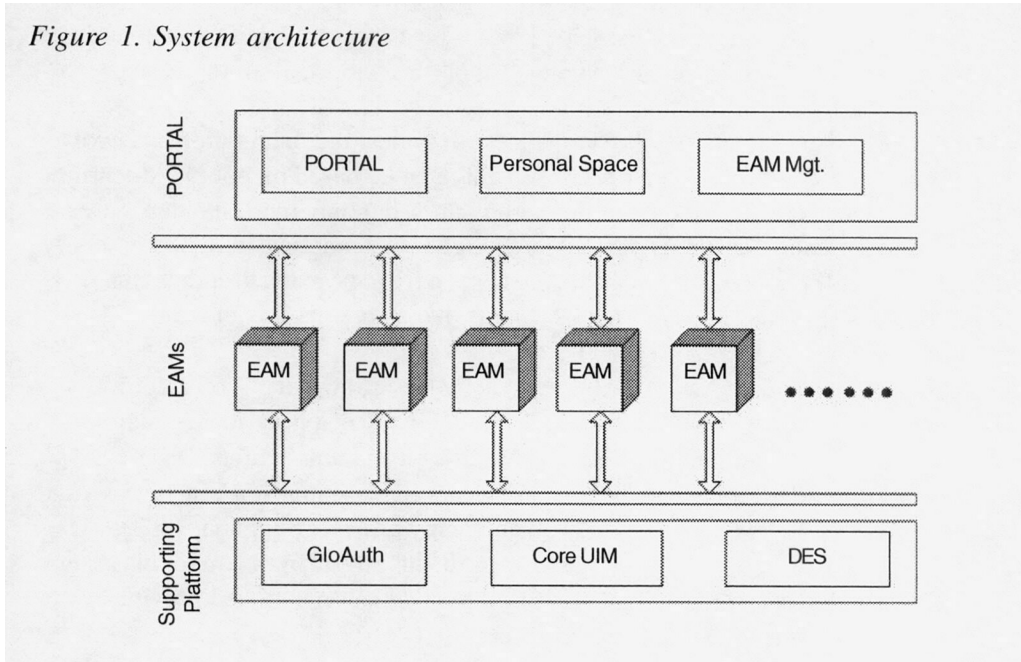
In this project, problems of integration and evolution mentioned previously are serious:

- Several e-learning standards — SCORM and CELTS (CELTSC, 2003), a Chinese localized version of selected standards from IEEE, AICC, ISO, and so forth — need to follow;
- Educational applications that developed into eight groups from different universities must be integrated into one virtual university;
- Some legacy applications need to be integrated into this virtual university;
- Evolution of these applications is probably frequent, as they will be used in different universities and modified according to the feedback.

We believe that an appropriate software architecture based on popular e-learning standards is promising to solve these problems. CUBES is the result of our efforts toward this direction.

In the following sections, CUBES is introduced in detail. The composition of CUBES, functions of important components, communications between these components, and design principles are dis-

Figure 1. System architecture



cussed. After that, a practical virtual university based on CUBES is discussed, proving the usability of CUBES. Finally, the conclusion and future work are given.

## SYSTEM ARCHITECTURE

### Overview

As depicted in Figure 1, CUBES is a three-layered architecture. The first layer is a universal portal. For students and teachers, it is the universal accessing entry; for system administrators, it helps to manage the underlying educational applications at runtime. The second layer is the application layer. There is only one kind of component in this layer, called EAM (Educational Application Module). EAMs are educational applications, such as courseware on-demand system, payment system, and learner management. They are supposed to be developed by different groups. The third layer is a supporting platform, providing common services for the upper EAMs and portal. The supporting

platform consists of three components: GloAuth (Global Authentication), CoreUIM (Core User Information Management), and DES (Data Exchange Server).

The purpose of adopting such a three-layered architecture is to simplify communications between different EAMs. As seen in Figure 1, there are no direct communications between EAMs; they only interact with the portal and the supporting platform. If one EAM is modified, most of the integration work is focused on the portal and the supporting platform, not bothering the other EAMs too much. Such an architecture greatly reduces the efforts in integration and evolvement, permitting designers and developers to build virtual universities with EAMs like piling cubes on the supporting platform. That is why it is called CUBES.

Comparing with other architectures for virtual universities such as OKI and IMS' Abstract Framework, CUBES provides less common services, because the data model of authentication and user information are common and mature in e-

learning standards, while other common services such as logging, file, and database manipulation are still not widely accepted. In CUBES, some of these common services are supposed to be implemented in EAMs, not in the supporting platform. These services interact with other EAMs through DES. Because modification in DES is much easier than that between EAMs, CUBES provides more flexibility.

In the following sections, EAM, GloAuth, CoreUIM, DES, portal, and communications between them are discussed. DES, as the most important component in CUBES, is described in more detail.

## EAM

There are many educational applications providing different kinds of functions for virtual universities. In order to integrate as many kinds of applications as possible in CUBES, the following assumptions for EAM are made:

- EAMs may have their internal database, user information management, authorization, and authentication;
- EAMs may be standardized modules or not; and
- Changing the internal data formats in EAMs is difficult.

These assumptions are reasonable. For example, a real-time virtual classroom is a complex system with its own data formats, usually incompatible with e-learning standards. In CUBES, EAMs are encouraged to preserve its own database, authentications, data formats, and so forth. Most of the integration efforts are moved into the communication interfaces between EAMs and the supporting platform; DES is used to solve the problem of data for-

mat. The necessity of additional authentication and user information management provided by GloAuth and CoreUIM is discussed later.

## Portal

The universal portal helps to provide a seamless learning environment for all of the users, including learners, teachers, and administrators. The portal determines how services provided by the underlying EAMs are presented. In the universal portal, users have a consistent view of integrated services without the concept of individual EAMs.

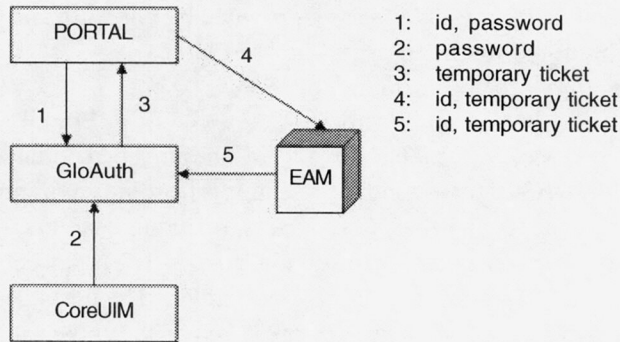
The portal consists of three components: course portal, personal space, and EAM management. Services from EAMs are grouped into different courses in the course portal. EAM management provides tools in order for administrators to configure the appearance of services (e.g., position on the Web page, relations with other services).

In practice, implementing such a universal portal remains hard work. In the future, a stable specification for the universal portal in virtual universities is required in the future.

## GloAuth

GloAuth provides a single sign-on service. A user only needs to login once in the universal portal, then he or she can navigate through all of the EAMs. The authorization process is depicted in Figure 2. When a user logs in, the portal sends ID and password to GloAuth. Comparing with the password from CoreUIM, GloAuth returns a temporary ticket to the portal. When the user navigates to an EAM, the temporary ticket and the user's ID are sent to the EAM automatically as the certification from the portal. The EAM asks GloAuth to validate the temporary ticket and let the user enter.

Figure 2. Communication of GloAuth



To keep security, the temporary ticket is updated after a period of time.

GloAuth is also responsible for authorization. It gathers required certifications from EAMs or the portal and manages a list for each user to store his or her authorization for EAMs. When the user navigates into an EAM, the EAM will query GloAuth for the user's authorization.

EAMs still may have their own internal authentication. There are, indeed, situations when the user needs to be authenticated more than once, except for the login in the portal. For example, the authentication in GloAuth relies on user ID and password, while the integrated payment system requires an X.509 certification for a higher level of security assurance. Preserving the internal authentications in EAMs guarantees the flexibility for multi-level security.

### CoreUIM

CoreUIM defines the data structures of learners' profiles and provides services for accessing and manipulating these profiles. At present, there is only one profile for each user in CoreUIM, called the core profile. The information stored in this profile is that defined in widely accepted e-learning

standards for user information, such as IEEE's PAPI and CELTS-11.

All of the EAMs share core user profiles. Meantime, an EAM may also maintain its own internal user profiles as depicted in Figure 3. Choosing such a strategy for managing user profiles is based on the following considerations:

- Some EAMs already have implemented their own user information management modules. It's troublesome to move these management modules into a global service.
- Some of the user information is EAM-specific, only used by the EAM internally. Separating the user information from CoreUIM helps to improve the extensibility.
- Some user information still lacks the corresponding e-learning standards. Not to put this information into the global user profile helps to improve the portability.
- Some EAMs have to keep their own user profile management; for example, the payment system for security considerations.
- Adopting the mature standardized user information into CoreUIM helps to reduce development efforts, as these

Figure 3. Hybrid user information

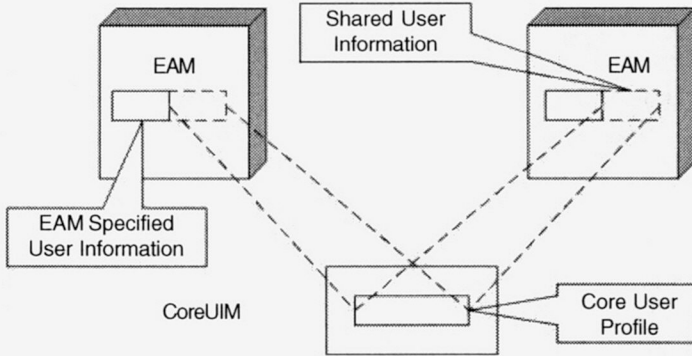
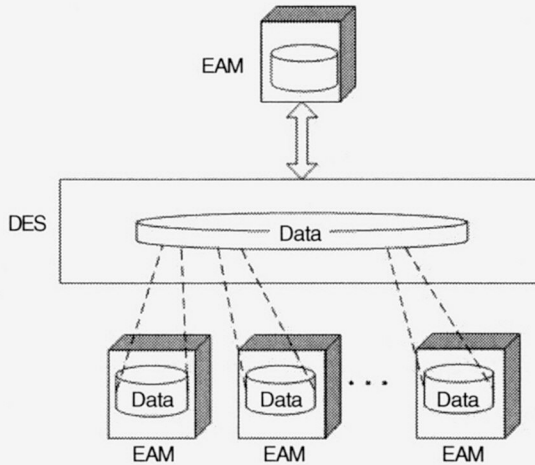


Figure 4. Abstraction of DES



data formats already have been widely accepted.

However, EAMs can exchange their user information not shared in CoreUIM through DES, if necessary.

**Data Exchange Server**

DES is the mediator for data exchanges between EAMs. As in Figure 4, DES acts as a virtual uniform data provider and consumer. There is no direct communication between EAMs; each EAM only communicates with DES, unaware of the existence of other EAMs.

Introducing DES between EAMs benefits from:

- Reducing the cost of integration. When an EAM is modified or a new EAM is added, most of the modification focuses on communications between this EAM and DES. Without DES, communications between the modified EAM and other EAMs have to be rewritten one by one.
- Being convenient to optimize the process of data exchange. Additional services such as cache can be introduced

Copyright © 2006, Idea Group Inc. Copying or distributing in print or electronic forms without written permission of Idea Group Inc. is prohibited.

into DES. Without DES, introducing such services usually leads to modification in every EAM.

- Having a centralized security model. It is convenient for DES to provide various communication protocols for security of the data exchange, such as HTTPS, to ease the implementation details of encryption and authentication.
- Having a runtime access control. The centralized DES allows the administrator to specify which EAM can participate, what data it can share, and what data from other EAMs it can access. Such a runtime access control is useful both in practice and in the testing phase.
- Easy-to-integrate EAMs developed in different languages. DES can implement different interfaces such as c/c++, Java, and Web service interfaces for EAMs. Without DES, integration of EAMs developed in different languages becomes much harder.

In the following section, design principles of DES are discussed in detail, which determines DES's interface and its internal structure. Then, the communication models between EAMs and DES are introduced. At last, DES's internal structure is described, including some important implementation details.

#### *Design Principles*

DEM is designed to provide stable uniform data exchange services, making the upper EAMs loosely coupling, while not losing too much efficiency at runtime. To meet the previous requirements, four design principles are concluded:

#### *REST-Like Interfaces*

REST (Fielding, 2000)-like interfaces are verb plus document style. In interfaces

of DES, four verbs are used: SEND, STORE, QUERY, and REGISTER. Documents appending to verbs are semantically rich and self-descriptive. When an EAM communicates with DES, it organizes the exchanged data into a self-descriptive document and tags the document with one of the verbs. As the number of verbs is very limited, interfaces based on these verbs are usually stable. Organization of documents in the sender EAM and explanation of the documents in the receiver EAM are both flexible. They can choose different schemas for documents as they will, because DES can adapt these documents when they are exchanged. REST-like interfaces prove flexible and are widely used for integration in many other systems.

#### *Stateless*

Communications between DES and EAMs are stateless. Every request from the EAM to DES (and vice versa) must contain all of the information necessary to understand the request, not taking advantage of any stored context on the receiver. That means session state is kept entirely on the client that sends the request.

The stateless feature improves visibility, reliability, and scalability (Fielding, 2000). When DES receives a request from the EAM, it does not have to look beyond the single request to determine the full meaning of the request. Recovering from partial failures becomes easier, which is important if EAMs distribute on the Internet. DES does not have to store states between requests, reducing the resource occupied by each request and, therefore, improving the number of requests supported by DES. The DES's ability of supporting a large number of requests makes significant sense in CUBES, because DES is the bottleneck for data exchanges.

### *Cache*

Cache takes a key role in DES to improve network efficiency. By utilizing cache in DES, it is possible to eliminate some communications between DES and EAMs, which means not only reducing communication latency but also saving network bandwidth.

However, not all kinds of exchanged data through DES are suitable for cache. When the data are stored in cache, DES must guarantee the consistency, which means additional communications for synchronization between DES and the EAM where the data come. If the data update frequently but are exchanged seldom, cache even reduces network efficiency.

In DES, the mechanism of cache is implemented, but it is the developer's duty to use cache properly in integration according to properties of each kind of data exchange.

### *Asynchronous Communication Model*

An asynchronous communication model means that when an EAM sends a request to DES, DES will synchronously acknowledge receipt of the request but is not sure to return the data requested immediately.

The synchronous communication model between DES and EAM is unreliable. In CUBES, EAMs may spread over a local network or the Internet. A synchronous model assumes that the network between EAMs and DES is reliable and behaviors of EAMs such as busy or not are predictable. For CUBES, a network-based architecture, these assumptions are inappropriate.

The asynchronous communication model ensures scalability and reliability. For example, in a data exchange process, when the EAM that provides data is busy or even unconnected due to the problem of network,

other EAMs can still send a request to DES, which will try to connect the data provider and return the data as soon as the provider is available.

### *Communications with EAMs*

Before an EAM communicates with DES, it must register the information about what data it can provide and what data it requires.

DES provides two asynchronous communication models: request-response model and publish-subscribe model.

In the request-response model, when an EAM requires data, it sends a request message to DES. DES acknowledges receipt of the request immediately and tries to collect the data from its own cache and other EAMs. First, DES looks into its own cache. If the data in cache is absent or out-of-date, it looks up registered EAMs for data providers and then sends a request to the provider, which is responsible to return the data. After collecting all of the required data, DES will examine the data format and call data adaptors to transform the data, if necessary. At last, DES sends the data to the EAM as a response. The exchange process is illustrated in Figure 5.

In the publish-subscribe model (Figure 6), if an EAM requires data, it subscribes through DES by sending an event that defines the boundary of the data required. Then DES looks up registered EAMs for data providers and subscribes. Other EAMs publish the required data back to DES, and then DES publishes data to the request EAM. The publish-subscribe model differs from the request-response model mainly in three points:

1. The subscription is persistent. If the EAM does not unsubscribe, it stays valid. As long as the subscribed data updates

Figure 5. Request-response model

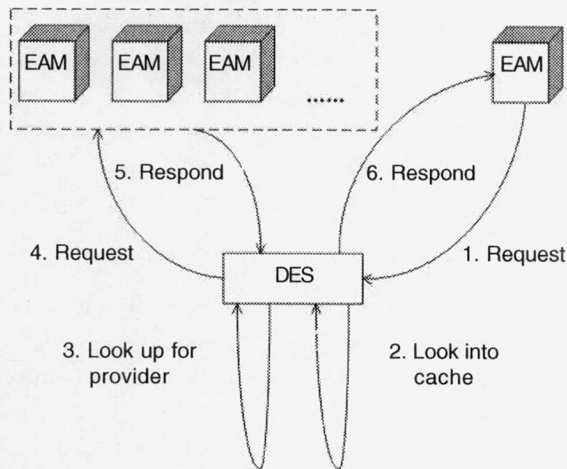
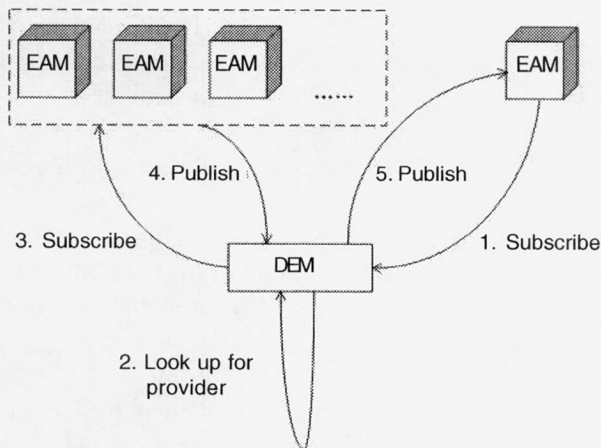


Figure 6. Publish-subscribe model



and the subscription remains valid, DES publishes data to the EAM. In the request-response model, there is only one response for one request.

2. The EAM publishes data to DES as soon as part of the subscribed data updates, and DES also publishes data to the EAM. In the request-response model, DES only sends data to the EAM when all of the requested data are ready.
3. In the publish-subscribe model, cache usually is not used.

Request-response model and publish-subscribe model are used in different conditions. If the exchanged data seldom update in the provider EAM, the request-response model is preferred. If the exchanged data update frequently, the publish-subscribe model is better.

The publish-subscribe model also is used for cache in DES. When DES makes a cache for some data, it must send subscriptions to EAMs that may modify these data. When these EAMs modify the data

Figure 7. Structure of DES

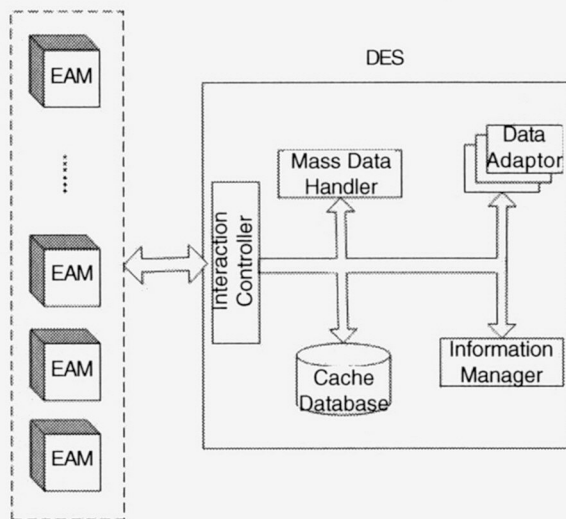
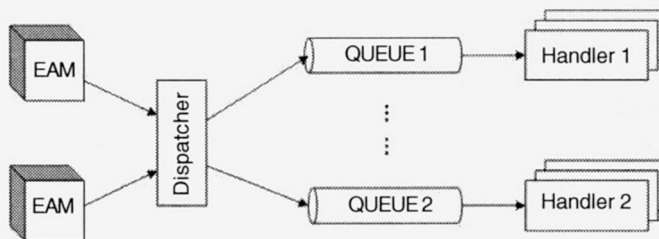


Figure 8. Interaction handler



cached in DES, it generates a message to notify DES so that DES can update the cache.

*Structure of DES*

As illustrated in Figure 7, DES consists of five components: Interaction Controller, Data Adaptor, Mass Data Handler, Cache Database, and Information Manager.

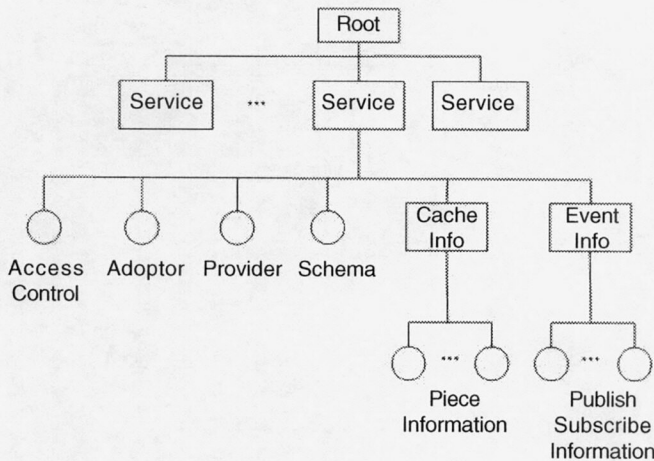
*Interaction Controller*

Interaction Controller implements all of the interfaces communicating with EAMs. It accepts requests and data from EAMs, calls the services provided by other components in DES, and responds to

EAMs. In order to support a large number of requests and to provide easy configuration for different requests, queues and events are utilized in Interaction Handler. As in Figure 8, when a request arrives, it is organized as an event and put into one of the queues in Interaction Handler according to properties of the request. Each queue corresponds to different priority and has its own handlers.

Multiple queues is better than one queue, because the number of initiated handlers for each queue is configurable at runtime, and it is easier to monitor it throughout for different kinds of requests in different queues.

Figure 9. Data structure of information management



### *Data Adaptor*

Data Adaptor consists of a series of adaptors; each of them transforms one data format into another. Implemented as Java Beans, adaptors can be modified or added into DES at runtime. Relationships between adaptors and data exchange processes are stored in Information Manager. When Interaction Controller initiates an exchange process, it looks up Information Manager for related adaptors and calls them.

### *Mass Data Handler*

Sometimes, there may be a large amount of data in one data exchange process. For example, when the learning quality monitor system requests logs of the articles posted on BBS in the last month, if the size of one log is 1 KB for 100,000 logs, the total size is 100 MB. If the two EAMs — BBS and the learning quality monitor system — do not locate in the same local network, sending so much data in one response from DES to EAMs is unreliable and costs too much in resources both in DES and EAMs. Opening a data socket for this kind of data exchange conflicts with predefined REST like interfaces; in addition,

when an EAM sends a request to DES, it does not know how much data it may receive.

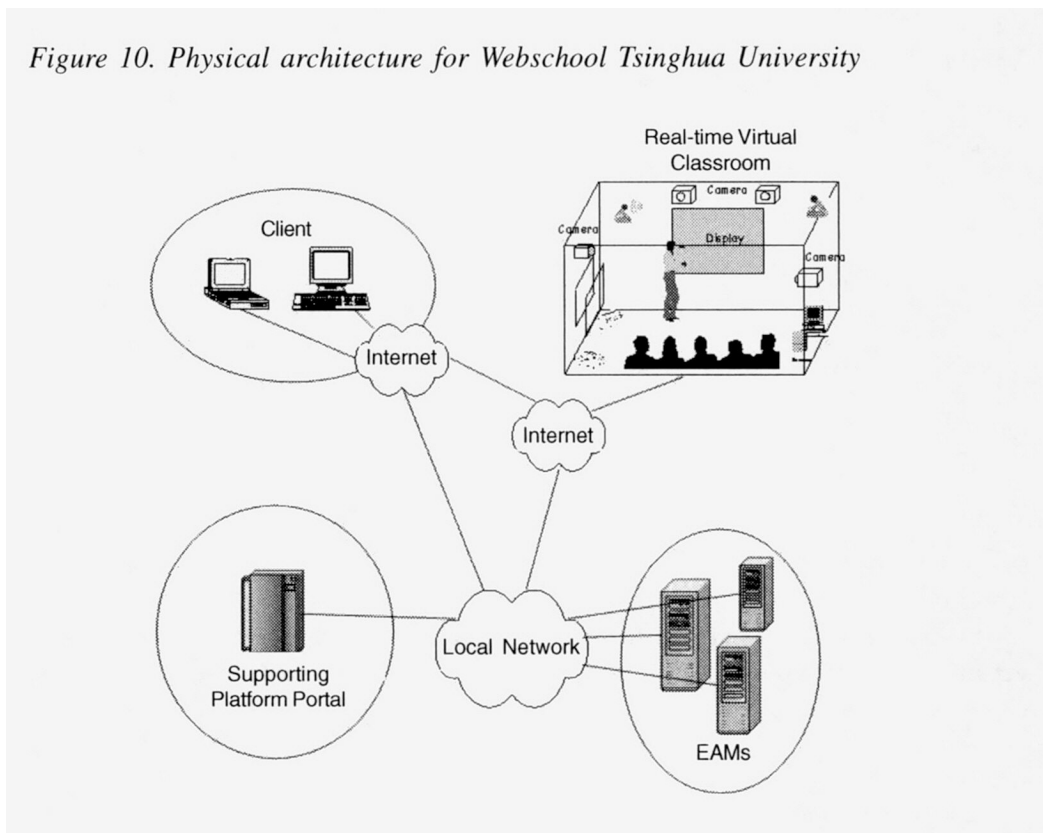
Mass Data Handler solves this problem by dividing the large amount of data into small pieces. When DES collects data from other EAMs, if the amount of data is beyond a predefined threshold, Mass Data Handler divides the data into small pieces and saves them in Cache Database; then, Interaction Controller responds to the EAM one piece of data for each time.

### *Information Manager*

Information Manager stores all of the information about the data exchange and provides services to access, modify, add, and delete this information. The information includes access controls that determine which EAM has the access authorization to which data, registered EAMs as data providers, relationships between adaptors and data exchange processes, schemas of the exchanged data, cache information, and publish-subscribe information.

The information is organized in a tree-based structure. As depicted in Figure 9,

Figure 10. Physical architecture for Webschool Tsinghua University



each service corresponds to a predefined data exchange process; other information is grouped according to these services. Information Manager is implemented on an LDAP server.

### A VIRTUAL UNIVERSITY BASED ON CUBES

Webschool Tsinghua University (Tsinghua, 2003), a virtual university regularly used by more than 10,000 users all over China, is constructed based on CUBES, providing about 150 online curriculum ranging from mathematics to computer technology, English, law, art design, and economics.

In this virtual university, a series of standardized or unstandardized EAMs developed by different organizations are integrated to provide a learning environment covering the entire spectrum of online learn-

ing and management processes. Some of the important EAMs are listed as follows:

- Large-scale, real-time interactive virtual classroom (Shi, 2003);
- Courseware-on-demand system;
- Resource management system for the metadata of learning materials;
- Quiz and test system;
- Payment system;
- Learner management system;
- Dynamic learning quality monitor;
- Assignment delivery and management system; and
- Question-and-answer system based on BBS.

Currently, the virtual classroom distributes on the Internet, while other EAMs locate in the same local network, connected

Table 1. Application of CELTS in EAMs

|                          | CELTS<br>-3 | CELTS<br>-11 | CELTS<br>-13 | CELTS<br>-9 | CELTS<br>-17 |
|--------------------------|-------------|--------------|--------------|-------------|--------------|
| Courseware on-demand     | ✓           |              |              | ✓           | ✓            |
| Resource Management      | ✓           |              |              | ✓           | ✓            |
| Payment System           |             | ✓            | ✓            |             | ✓            |
| Learner Management       |             | ✓            | ✓            |             | ✓            |
| Learning Quality Monitor |             | ✓            | ✓            |             | ✓            |
| Assignment               | ✓           |              |              | ✓           | ✓            |
| Question and Answer      | ✓           |              |              | ✓           | ✓            |
| Quiz and Test            | ✓           | ✓            |              | ✓           | ✓            |

together by the supporting platform (Figure 10).

In the practical use of the virtual university, some problems are found in CUBES:

- If an EAM connects to the supporting platform through the Internet, and if there is a large amount of data exchanged between DES and the EAM, the architecture of CUBES is inappropriate, because the latency of the data exchange heavily reduces the usability of the EAM.
- If the exchanged data require strict synchronization between the provider and the receiver, communications between DES and EAMs increase rapidly. If there is a large amount of data that needs to be strictly synchronized, the whole system becomes unreliable.

In essence, CUBES spends the efficiency at runtime in return for the efficiency in development. It is suitable for the integration of EAMs coming from different providers and for virtual universities that need to evolve frequently.

EAMs in Webschool Tsinghua University follow e-learning standards of CELTS and SCORM. Table 1 lists the

CELTS standards implemented in EAMs. SCORM (version 1.2) is applied in the courseware-on-demand system and the resource management system.

## CONCLUSION AND FUTURE WORK

This article introduces CUBES, a new software architecture for virtual universities that focuses on the integration and evolvement of EAMs developed by different organizations. CUBES is a three-layered architecture including a universal portal, collections of EAMs, and a supporting platform. By utilizing data exchange services, global authentication, and core user information management service in the supporting platform, the following characteristics are achieved:

1. Reducing the cost of integration;
2. Optimizing the process of data exchange;
3. A centralized security model;
4. Easy-to-integrate EAMs developed using different languages;
5. A flexible user information model; and
6. A unified user access entry.

For DES, the most important component in CUBES, design principles are concluded, which guide implementations of communication models and the internal structure of DES.

A regularly used virtual university, WebSchool Tsinghua University, which is based on CUBES, is introduced to examine the usability of CUBES. It proves that CUBES is suitable for integration and involvement for EAMs but not good under some conditions. In essence, CUBES spends the efficiency at runtime in return for the efficiency in development.

In the future, CUBES will be modified to facilitate the integration of EAMs that distribute over the Internet, so that EAMs running in different universities and institutions can be organized together to provide unified distance learning services.

## REFERENCES

- ADL. (2001). *SCORM Version 1.2*. Retrieved from <http://www.adlnet.org>
- Blackboard. (2004). Retrieved from <http://www.blackboard.com>
- CELTSC. (2003). *China e-learning technology standardization*. Retrieved from <http://www.celtsc.edu.cn>
- CMU. (2001). *Learning service architecture*. Retrieved from <http://www.lsal.cmu.edu>

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures*. Doctoral dissertation, University of California Irvine, Irvine, CA.

IEEE LTSC. (2001). *IEEE LTSA Working Draft 9*. Retrieved from <http://ltsc.ieee.org/wg1>

IMS. (2003). *IMS abstract framework Version 1.0*. Retrieved from <http://www.imsglobal.org>

OKI. (2004). *OKI open service interface definitions Version 2.0*. Retrieved from <http://web.mit.edu/oki>

Shi, Y., Xie, W. Xu, G., Xiang, P., & Zhang, B. (2003). Project smart remote classroom providing novel real time interactive distance learning technologies. *International Journal of Distance Education Technologies*, 1(3), 28-45.

Tsinghua. (2003). *Webschool Tsinghua University*. Retrieved from <http://www.itsinghua.edu.cn>

WebCT. (2004). Retrieved from <http://www.webct.com>

## ENDNOTE

- <sup>1</sup> This article is the extended version of the preliminary paper, "CUBES: Providing Flexible Learning Environment for Virtual Universities," which appeared in *ICWL-2004 Conference Proceedings*.

*Peifeng Xiang (xpf97@mails.tsinghua.edu.cn) is a PhD candidate of the Department of Computer Science, Tsinghua University. He researches on human-computer interface in pervasive computing and e-learning systems. He got his BSc from the Department of Computer Science, Tsinghua University in 2001.*

*Yuanchun Shi (shiyc@tsinghua.edu.cn) received her PhD, MSc, and BSc from the Department of Computer Science, Tsinghua University. She is now a professor of the Department of Computer Science, Tsinghua University. Her research interests include pervasive computing, distributed multimedia, and e-learning technology. She was also a senior visiting scholar at MIT AI lab during 2001-2002.*

*Weijun Qing (qinweijun99@mails.tsinghua.edu.cn) is a PhD candidate of the Department of Computer Science, Tsinghua University. He researches on context-aware computing and e-learning systems. He got his BSc from the Department of Computer Science, Tsinghua University in 2003.*