Overall Framework

We are building a new generation of distance education technology where we intend to re-use technologies aimed at areas like e-commerce and commodity Web resources. We will build on the emerging integration of distributed, component, and Web technology with our approach being compatible with the many competing candidates for the base infrastructure. We are working with the NCSA Alliance within an architecture that can be applied to both computing and education portals. We are considering Ninja from UC Berkeley and E-Speak from Hewlett Packard as interesting new approaches, and we will evaluate these over the summer as a possible infrastructure for this project. We also see some analogies between the requirements for a learning environment and the successful but controversial Gnutella or Napster type distributed archive technology for multimedia material.

We will apply the new system both to an existing network of historically black colleges and universities to Florida State's major institutional effort in distance and distributed learning. Fox brings his research effort from Syracuse University in this area and he will be chief technologist of FSU's ODDL (Office of Distributed and Distance Learning). Together these will cover undergraduate, graduate and lifelong learners. Further information can be found at:

http://www.new-npac.org/users/fox/hp/draftapril8.html and
http://www.new-npac.org/users/fox/documents/pajavaapril00/

To ensure that we can protect our investment we will adopt well-defined interfaces implemented in terms of XML and if necessary change our implementation as technology evolves. We use a 3-tier architecture with client, server and backend resource and the two interfaces, as shown in Fig. 1. This approach has been adapted successfully in the Gateway Web based computing project [http://www.osc.edu/~kenf/theGateway/ and http://www.npac.syr.edu/users/haupt/WebFlow/] with the use of two interfaces separating the user and system object view and insulating both the user interface and repository resources from the changing server infrastructure. As a simple example from the relational database field, resourceML would define the table structure used to classify the data while portalML would support user queries in SQL. Our application of this to distance education is detailed later in fig. 2 and the backend includes the courseware as well as the events (information nuggets) describing the users and their interactive sessions. Our proposed system will support the courseware developer who is adding or editing modules as well as the learners and teachers accessing the courseware repository. In addition it will provide tools to support person to person and person to database interactions. Existing standards efforts (IMS, ADL, IEEE LTSC) have provided a good start to these interfaces although they base on a less sophisticated client server model and essentially merge these two interfaces. In the following section, we elaborate our technical approach built around the concept of a collaborative portal.

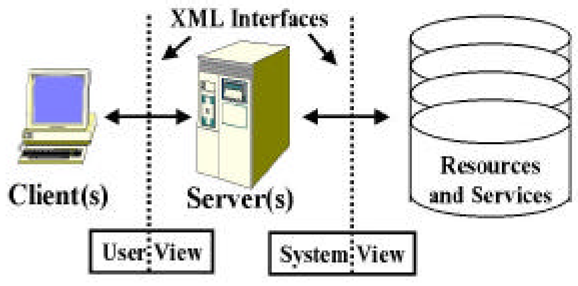

Fig. 1: Learning System Architecture with two Interfaces. User View (portalML) and System View (resourceML)

Collaborative Portals

It is unrealistic today for any one to build a complete online education environment from scratch: rather one must integrate a system from a variety of different sources. This motivates the standards for re-usable objects described in the previous section.

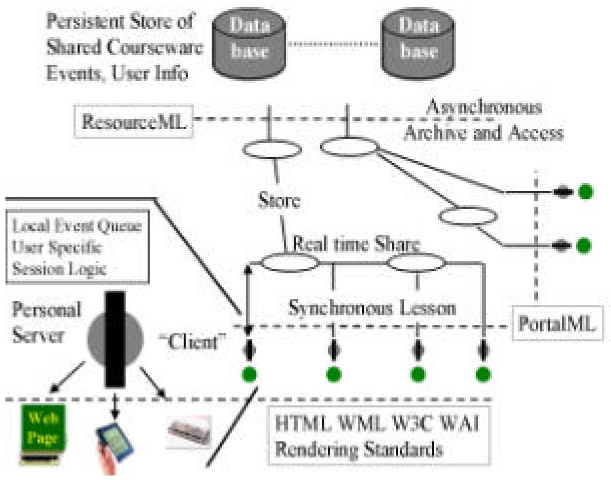

Fig. 2: Collaborative Portal showing support for multiple user interfaces and the event queue shared synchronously as well as being stored for asynchronous access

In this project we take an approach that in modern parlance is called an *educational portal.* A portal employs a modern distributed object framework and uses it to support distributed learning objects and services with the two interfaces defined above. We adopt a layered approach with one set of capabilities common to all portals and then specialize to different applications. Here we view a portal as "just" a web interface to a particular application area.
The general properties of any portal include storing, accessing and searching for distributed objects (which of course include web pages) in a repository. Further we have general services such as security and collaboration where the latter is particularly important for education as it enables the synchronous or asynchronous interactions between students and teachers. Further general portal capabilities include layout (of the rendered objects on a page), provision of metadata, universal access, user customization and performance (through use of mirror or proxy servers). We will research the use of the client-server interface (see Fig. 1) to define the object properties of relevance to these functions and as usual express them in terms of XML as "portalML". As shown in the SCORM standard for education objects from ADL (http://www.adlnet.org/ADL-TWG/documents.htm), one must support both base educational objects (modules) and their integration into lectures, courses, curriculum etc. We did this with our early WebWisdom system and an attractive interface for this can be seen in commercial software such as RealJukebox, which is designed to collect multimedia objects, which are simpler but have interesting points in common with learning objects. This software also supports neat layout customization through different "skins".

Returning to education, one must support special services such as assessment, performance (grading) support, and annotation. There are also distinctive "educational objects" – quizzes, homework, glossaries as well as the curriculum pages with appropriate hierarchical structure. Here we will extend SCORM and IMS but separate the "user view" from the basic resource specification. The latter ("system view") describes the learning modules stored in a shareable courseware repository (see back end in Fig. 2) We will of course pay attention to support for key capabilities such as displaying mathematics and other symbolic notations on the Web as well as standards for graphics (Java3D, VML, X3D etc.). This distributed object based system will have to support curriculum material built in any web authoring system and specified either statically or dynamically (from a database). This simple request turns into a serious challenge, as it requires the unification of services such as those for customization, collaboration, and events. This is a key research area as such unified services are essential for the basic strategy of allowing components from multiple academic and commercial sources. A simpler version of this challenge is well-defined XML interfaces to allow interoperability of data streams.

This appears a complex daunting agenda but fortunately many of the capabilities are provided by the new generation of Internet infrastructure such as E-speak. Therefore for this project we can focus on a few key issues. We will assume that new browsers (Internet Explorer 5 and Netscape 6) will have satisfactory support for the W3C document object model and XML. This already provides a nice way of specifying collections that is consistent with ADL's SCORM. We will build some simple layout tools supporting a portalML allowing natural grid and flow layouts (using a Java AWT notation). We assert that that key new capability shown in fig. 2 is an event service that allows one to receive and send time-stamped tagged messages. These events define the state of each portal page and can be used to support user customization by saving the event queue. The event queue is designed as a distributed (XML) database to support guarantees of robust delivery and performance through replication of shared events. The event log can also be used in assessment of both the student and the learning material as it records the user's interactions with the environment. As discussed in the Syracuse theses of Lee and Sen (students of Fox), this can be done server side when it reduces to the classic analysis of Web Server accesses logs. More interesting is the tracking of client side events where the challenge is basically datamining user relevant information. We will on one hand build in support for this as part of our event service and research extensions of the simple analyses in the two theses to automatically derive user profile and learning assessment information. This client side event information can be used to support universal access as described by Fox and Gilman from the Wisconsin Trace center (http://www.npac.syr.edu/users/gcf/montrealxmlaug99).

Our web-based virtual university approach implies that collaboration is a service that provides the sharing of web-based distributed objects. Previous systems have tended to support either synchronous or asynchronous collaboration modes, but based on our current experience we will unify them for this proposal. Initial synchronous deliveries have had some success using systems like Microsoft NetMeeting, NCSA's Habanero, and Syracuse's TangoInteractive. However the new requirements imply we will build collaboration in terms of the event service of our E-Speak framework. We will allow this to support either synchronous delivery or event archiving and later delivery of a session. Session control will be implemented in XML using the generalized portalML described above. We have found that developing shared animations (for education) is too difficult in current systems like TangoInteractive, which only support complex collaboration-aware applications without difficulties. We will use VNC or an equivalent technology to allow both shared display and collaboration-unaware applications, which are less flexible but much easier to author. One important issue of our research will be the techniques needed to provide this unified approach to collaboration. We are already building examples of this architecture shown in fig. 2, with an event service, which is designed to support the performance of immediate forwarding of object state changes that is needed by synchronous collaboration. This is combined with the archiving of events to support later asynchronous browsing of the course by users accessing the persistent database. We ran in difficulties with TangoInteractive due to its extensive use of browser-based software. In this approach we will avoid putting significant client side logic into a browser but rather use a "personal server". Here we view the browser (on a PC or hand-held device) as one particular rendering device – it contains the code to support rendering but the session logic and important data is controlled client side by a server. This approach allows a single user session logic to support multiple display devices including cross disability access such as a pure audio rendering for the visually impaired.

One continual area of challenge is the variable quality in digital audio and video conferencing. Higher speed in networking and improving quality of service will address some of the difficulties. We will track the ANL/NCSA Access Grid project at the high end, but for many educational uses commercial systems like RealAudio/Video can be used. In our multi-paradigm framework, we will allow the user to switch dynamically between interactive audio-video technology and the more reliable non real time systems (like RealAudio) whose larger buffer sizes are less sensitive to the lack of quality of service on today's internet. We have noted in our classes between JSU and Syracuse that we could use the more robust approach when the teacher is lecturing and interacting with the class through the chat rooms rather than the audio channel. This accounts for well over 95% of the time of a typical lecture.

We intend that a prototype system be available in spring 2001 and we will start with CORBA technology of Gateway adding the new event service as the first key enhancement. Then during fall of 2000 we will begin use of E-speak and Ninja for the new system after evaluating them over the summer.