KnowledgeTree:
A Distributed Architecture for Adaptive E-Learning

Peter Brusilovsky

School of Information Sciences, University of Pittsburgh
Pittsburgh PA 15260
+1 412 624 9404
peterb@mail.sis.pitt.edu

ABSTRACT

This paper presents KnowledgeTree, an architecture for adaptive E-Learning based on distributed reusable intelligent learning activities. The goal of KnowledgeTree is to bridge the gap between the currently popular approach to Web-based education, which is centered on learning management systems vs. the powerful but underused technologies in intelligent tutoring and adaptive hypermedia. This integrative architecture attempts to address both the component-based assembly of adaptive systems and teacher-level reusability.

Categories and Subject Descriptors

H.4 [**Information System**]: Information System Applications. K.3.1 [**Computers and Education**]: Computer Uses in Education - *Distance Learning.*

General Terms

Management, Design, Human Factors, Standardization.

Keywords

Adaptive Web, E-Learning, Learning Portal, Adaptive Content Service, Student Model Server, Learning Object Metadata, Content Re-use.

1. INTRODUCTION

The technological landscape of modern E-Learning is dominated by so-called learning management systems [10] such as Blackboard [4] or WebCT [42]. Learning management systems (LMS) are powerful integrated systems that support a number of activities performed by teachers and students during the E-Learning process. Teachers use an LMS to develop Web-based course notes and quizzes, to communicate with students and to monitor and grade student progress. Students use it for learning, communication and collaboration. As is the case for a number of other classes of modern Web-based systems, LMS offer their users "one size fits all" service. All learners taking an LMS-based course, regardless of their knowledge, goals, and interests, receive access to the same educational material and the same set of tools, buffered with no personalized support.

Adaptive Web-based Educational systems (AWBES), a recognized class of adaptive Web systems [9] attempt to fight the "one size fits all" approach to E-Learning. After almost 8 years of research on adaptive E-Learning, this field can demonstrate some impressive results [6]. For every function that a typical LMS performs we can find a number of AWBES that can do it much better than the state-of-the-art LMS. Adaptive textbooks created with such systems as InterBook [8], NetCoach [44] or ActiveMath [29] can help students learn faster and better. Adaptive quizzes developed with SIETTE [34] evaluate student knowledge more precisely with fewer questions. Intelligent solution analyzers [43] can diagnose solutions of educational exercises and help the student to resolve problems. Adaptive class monitoring systems [32] give the teachers a much better chance to notice when students are lagging behind. Adaptive collaboration support systems [37] can enhance the power of collaborative learning.

The traditional problems involved in authoring adaptive learning content have been nearly resolved by the new generation of powerful authoring tools. Authoring support in modern AWBES such as NetCoach [44] or SIETTE [34] is comparable with modern LMS. Moreover, a number of existing AWBES are provided with a wealth of existing or newly created learning materials, while the typical LMS expects teachers to develop all learning materials themselves. For example, ELM-ART [43] comprehensively supports the most important portions of a typical Lisp course - from concept presentation to program debugging. Yet, seven years after the appearance of the first adaptive Web-based educational systems, just a handful of these systems are actually being used for teaching real courses, typically in a class lead by one of the authors of the adaptive system.

The problem of the current generation of AWBES is not their performance, but their architecture. Structurally, modern AWBES do not address the needs of both university teachers and administration. *The first issue is the lack of integration.* While AWBES as a class can support every aspect of Web-enhanced education better than LMS, each particular system can typically support only one of these functions. For example, SIETTE [34] is a great system for serving quizzes, but it can't do anything else. To cover all needs of Web-enhanced education with AWBES, a teacher would need to use a range of different AWBES together. This is clearly a problem for the university administration that is responsible to maintain and provide training for all these systems It is also a burden for the teacher who needs to master them all and for the student who needs to manipulate several systems and interfaces - all with separate logins – and all at the same time. E-Learning stakeholders have a clear need for a single-entrance, integrated system that can support all critical functions in one package. LMS producers have recognized this need several years ago. Just in a few years after their emergence, LMS have progressed from one-or-two function systems into Web-based monsters that can cover all needs.

The second issue is the lack of re-use support. Modern AWBES are self-contained systems and can't be used as components. A teacher who is interested in re-using some

WWW 2004, May 17–22, 2004, New York, New York, USA.
ACM 1-58113-912-8/04/0005.

adaptive content from an existing adaptive system (for example, several ELM-ART Lisp problems) has only one choice - to accept all or none of an intact system, with its specific way of teaching, thereby sacrificing his or her preferred way of teaching the course. Once one excludes the *authors* of existing adaptive systems who built those systems to support *their* way of teaching, it is rare that one finds a teacher who is willing to do that. In contrast, LMS have always supported teachers in developing their course material from various components. Modern courseware-reusability frameworks such as ARIADNE [41] extend this power by providing repositories of reusable educational objects.

The author is convinced that the way to E-Learning classroom for AWBES systems goes through a component-based architecture for adaptive E-Learning. After exploring several approaches for building a distributed component-based architecture [5; 12], we have developed KnowledgeTree, a distributed architecture for adaptive E-Learning based on reusable intelligent learning activities. KnowledgeTree addresses both the component-based development of adaptive systems and the teacher-level reusability. It attempts to bridge the gap between powerful but underused AWBES technologies and the modern approach to Web-based education based on LMS and repositories of educational material. The first version of the architecture was implemented at the end of 2001 [11]. Since the Spring 2002 semester, KnowledgeTree has been used to support several courses at the University of Pittsburgh. Over time we have added and redesigned several components and refined the architecture itself. We have gained some solid practical experience with the distributed architecture and have had a chance to compare it with other solutions. In this paper we present the architecture, describe its most recent version, review our experience, and analyze several problems that have to be addressed in the future work.

2. THE ARCHITECTURE

KnowledgeTree is a distributed architecture for adaptive E-learning based on the re-use of intelligent educational activities. Capitalizing on the success of integrated LMS, KnowledgeTree aims to provide one-stop comprehensive support for the needs of teachers and students who are using E-Learning. It doing so it attempts to replace the current monolithic LMS with a community of distributed communicating servers (or services). The architecture assumes the presence of at least four kinds of servers: activity servers, value-adding services, learning portals, and student model servers (Figure 1). These kinds of servers represent the interests of three main stakeholders in the modern E-Learning process: content and service providers, course providers, and students.

A *learning portal* represents the needs of course providers - teachers (trainers) and their respective universities (or corporate training companies). A portal plays a role similar to modern LMS in two aspects. First, it provides a centralized single-login point for enrolled students to work with all learning tools and content fragments that are provided in the context of their courses. Secondly, it allows the teacher responsible for a specific course to structure access to various distributed fragments according to the needs of this course. Thus, a portal is a component of the architecture that is centered on supporting a *complete course*. Replicating the familiar functionality of an LMS, it provides a course-authoring interface for the teacher and maintains a runtime interface for the student. The difference between this and LMS an architectural separation of the unique *course structure* created by the teacher or course author from reusable course content and services. In KnowledgeTree, both learning content and learning support services (together called *activities*) are provided through the portal by multiple distributed *activity servers* (services). A portal has the ability to query activity servers for relevant activities and launch remote activities selected by students or by the portal itself.

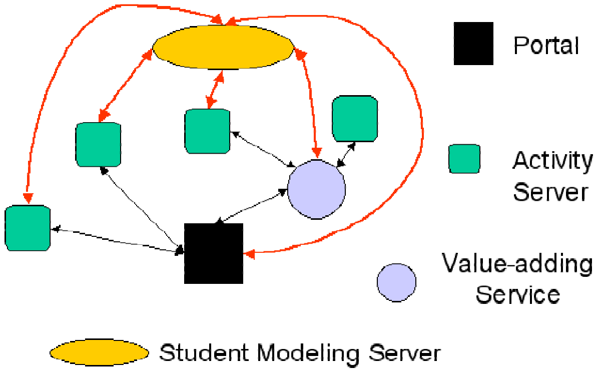


Figure 1. Main components of the KnowledgeTree distributed architecture.

An *activity server* is a component that focuses on the prospects needs of content and service providers. It is centered on reusable *content and services*. It plays a role similar to an educational repository in modern courseware reusability approaches, in the sense that it hosts reusable learning content. The difference between this and a traditional learning repository is twofold. First, unlike repositories that are pools for storing simple, mostly static, learning objects, an activity server can host highly interactive and adaptive learning content. It can also host interactive learning services such as discussion forums or shared annotations. Secondly, an activity server assumes a different way of re-using its "content". While simple learning objects are re-used by being copied and inserted into new courses, an activity is re-used by referencing and is then delivered by its server.

The need for activity servers stems from the nature of adaptive and other advanced learning activities - such as ELM-ART problems. These activities just can't be copied as files, they have to be served from a dedicated Web servers maintained by the content providers. The duty of an activity server is to answer the portal's and value-adding service's requests for specific activities and to provide complete support for a student working with each of the activities residing on the server. The concept of reusable activities encourages content providers to develop highly advanced, interactive learning content and services. In particular, content and service activities delivered by a server can be intelligent and adaptive. Each activity can obtain and up-to-date information about each student from the student model server and thus provide a highly personalized learning experience. It also monitors student progress, changes student goals, knowledge, and interests; then sends updates to the student model server.

A value-adding service combines features of a portal and an activity server. It is able to "pass through" itself the "raw" content and services adding some valuable functionality to it - such as adaptive sequencing, annotation, visualization, or content integration. Like a portal, it is able to query activity servers and access activities. Like an activity server, it can be queried and accessed by a portal. Value-adding services are maintained by service providers. Since these services are course-neutral, they can be re-used in multiple courses providing larger building blocks for a teachers assembling an E-Learning system with the help of a portal.

The *student model server* is a component that represents the needs and the prospects of students in the process of E-Learning. This kind of server allows distributed E-Learning to be highly personalized. Ideally, a student model server can support student learning for several courses. It can be maintained by a provider (i.e., a university) or by the students themselves. It collects data about student performance from each portal and each activity server and provides information about the student to adaptive portals and activity servers that are then able to adapt instructional materials to their students' unique personalities and present development.

We argue that the presence of multiple adaptive activities requires a centralized user modeling architecture that enables each learning activity to get access to all information about student progress. The problem of centralized student modeling servers has been investigated in a number of earlier projects dealing with multiple intelligent educational agents [5; 28]. Also, the Web context poses new requirements for student model servers. Our server CUMULATE, presented below, and the Personis server, suggested by Kay, Kummerfeld, and Lauder [26] provide examples of student model servers that can be used in distributed adaptive E-Learning environments.

We anticipate that in the context of pure Web-based education, a student model server can reside on the student's own computer and support just one user. Using this method, the server can also serve as a tool for the user to monitor his or her own progress within various activities and courses. In the context of classroom education, the server can reside on a computer maintained by the educational establishment. Here it also supports the teacher's need to monitor the progress and the performance of the whole class. These arrangements can help to solve a number of privacy and security problems associated with student modeling.

With the KnowledgeTree architecture, a teacher develops a course using one portal and many activity servers and services. The student works through the portal serving this course, but interacts with many learning activities, served directly by various activity servers. The student model server provides a basis for performance monitoring and adaptivity in this distributed context. The KnowledgeTree architecture is open and flexible. It allows the presence of multiple portals, activity servers, and student modeling servers. The open nature of it allows even small research groups or companies to be "players" in the new E-learning market. It also encourages creative competition between developers of educational systems - i.e., competition based on offering better services, not by monopolizing the market and resisting innovation. An activity server that provides some specific innovative learning activities can be immediately used in multiple courses served by different portals. A newly created portal that offers better support for a teacher or answers better to the needs of a specific category of course providers can successfully compete with other portals since it has access to the same set of resources as other portals. An new kind of student model server that provides better precision in student modeling or offers a better support for student model maintenance can successfully compete with older servers. Overall the architecture reflects the move from a product-based to a service-based Web economy.

The open nature of the architecture relies on several clearly-defined communication protocols between the components. First of all, the architecture needs a protocol for transparent login and authentication. Each adaptive activity should know the identity of the student in order to be able to communicate with the student model server; however the student should log in only once when starting a new session with the portal. Secondly, a protocol is required for communication with activity servers. Since each activity server becomes a small pool of educational content and services, it should be able to answer content queries – listing search results: activities that match a specific description (in terms of metadata) or provide all known metadata for a specific activity. It should also have the ability to launch an activity by direct request. Thirdly, a protocol is needed for the activity server or portal to send the information about the student progress to the student model server and as well as a protocol to request information about the student from the student model. Finally, the architecture needs a mechanism for resource discovery and exchange. A portal can provide an access to a wide variety of learning activities residing on many servers. However, to benefit from this feature, a portal should be aware of many independent servers and the kinds of activities they can offer.

Our current implementation of KnowledgeTree features a simple implementation of the first three protocols. At the end of the next section we review this implementation and discuss possible alternatives.

3. THE IMPLEMENTATION

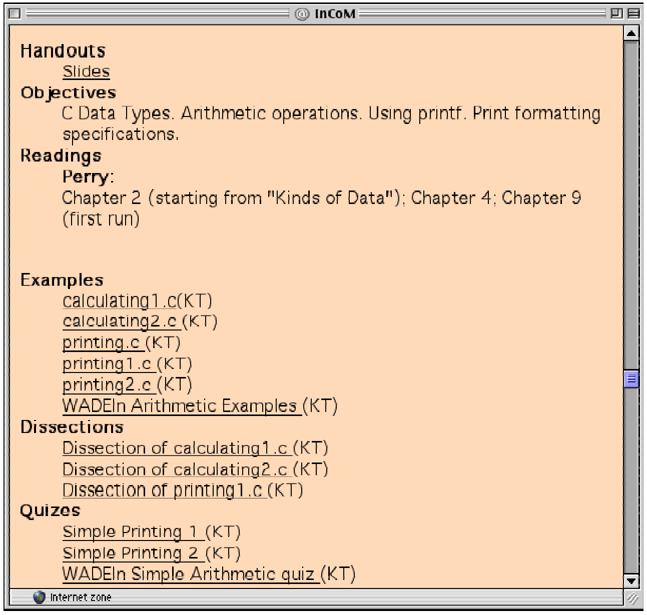
3.1 The Portals

The KnowledgeTree architecture allows multiple portals that can support different educational paradigms and approaches while providing access to the same universe of distributed content and services. So far we have implemented two very different portals. The first portal that we developed was targeted to support a traditional lecture-based educational process in a university. It was named KnowledgeTree (eventually providing the name for the overall architecture) since it supported a teacher in structuring a Web course as a tree (or sequence) of learning units. KnowledgeTree supports both lecture-based and textbook-based course organization. With this portal, a teacher can define a sequence of lectures or a tree of sections, then add primary and secondary learning material for each lecture or section. Thus, KnowledgeTree supports a course-authoring approach that should be familiar to all course authors currently using modern LMS (Figure 2).

The learning material itself - both static and interactive - is not stored on the portal. Static material is served from Web servers or through an annotation service called AnnotatED, while interactive material is served from activity servers. The current version of KnowledgeTree is *not* student-adaptive and does not even query the student model server. We are currently developing a new version of KnowledgeTree that will be based on adaptive hypermedia technologies.

The Knowledge Sea portal [13] is dramatically different from KnowledgeTree. It represents our attempt to meaningfully structure large volumes of learning material with minimal human involvement. With Knowledge Sea, the only duty of a teacher is to specify the range of learning material (activities) to be used in the course. For example, the teacher of a C programming course can pass to Knowledge Sea a set of lecture notes along with the links to several Web-based C tutorials containing dozens to hundreds of content pages each.

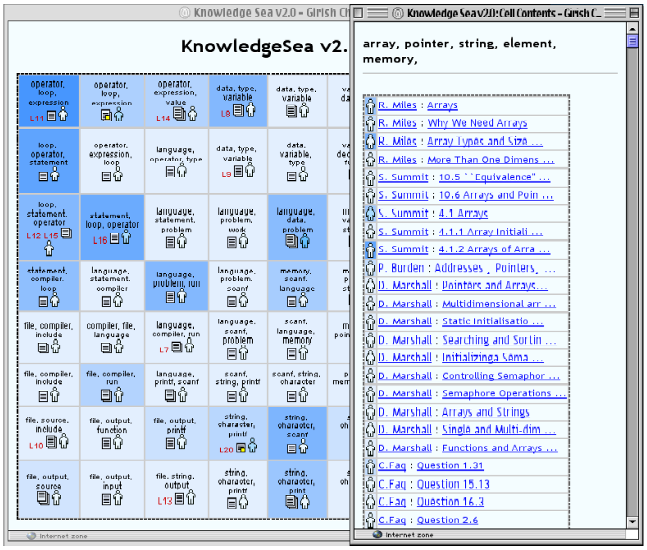
Knowledge Sea uses information retrieval technologies and self-organized neural networks to structure all provided content as a matrix-based knowledge map. Currently we use 8x8 maps. Each cell in the matrix gathers links to semantically similar learning resources. In addition, the content of connected cells is also relatively similar (thus keeping the topology of the represented knowledge space intact).


InCoM

Handouts
Slides
Objectives
C Data Types. Arithmetic operations. Using printf. Print formatting specifications.
Readings
Perry:
Chapter 2 (starting from "Kinds of Data"); Chapter 4; Chapter 9 (first run)

Examples
calculating1.c(KT)
calculating2.c (KT)
printing.c (KT)
printing1.c (KT)
printing2.c (KT)
WADEIn Arithmetic Examples (KT)
Dissections
Dissection of calculating1.c (KT)
Dissection of calculating2.c (KT)
Dissection of printing1.c (KT)
Quizes
Simple Printing 1 (KT)
Simple Printing 2 (KT)
WADEIn Simple Arithmetic quiz (KT)

Figure 2. A user view of a lecture in the KnowledgeTree portal showing links to various external learning activities


Knowledge Sea v2.0 - Girish Ch

KnowledgeSea v2.

operator, loop, expression L11	operator, loop, expression	operator, expression, value L14	data, type, variable L8	data, type, variable
loop, operator, statement	operator, expression, loop	language, operator, type	data, type, variable L9	data, variable, type
loop, statement, operator L12 L15	statement, loop, operator L16	language, statement, problem	language, problem, work	language, data, problem
statement, compiler, loop	language, statement, compiler	language, problem, run	language, problem, scanf	memory, scanf, language
file, compiler, include	compiler, file, language	language, compiler, run L7	language, scanf, problem	scanf, language, memory
file, compiler, include	file, compiler, run	language, printf, scanf	scanf, string, printf	scanf, string, character
file, source, include L10	file, output, function	file, output, printf	string, character, printf L20	string, character, scanf
file, output, source	file, output, input	file, string, output L13	string, character, printf	string, character, print

Knowledge Sea v2.0: Cell Contents - Girish C

array, pointer, string, element, memory,

R. Miles : Arrays
R. Miles : Why We Need Arrays
R. Miles : Array Types and Size ...
R. Miles : More Than One Dimens ...
S. Summit : 10.5 ``Equivalence" ...
S. Summit : 10.6 Arrays and Poin ...
S. Summit : 4.1 Arrays
S. Summit : 4.1.1 Array Initiali ...
S. Summit : 4.1.2 Arrays of Arra ...
P. Burden : Addresses , Pointers, ...
D. Marshall : Pointers and Arrays...
D. Marshall : Multidimensional arr ...
D. Marshall : Static Initialisatio ...
D. Marshall : Searching and Sortin ...
D. Marshall : Initializinga Sema ...
D. Marshall : Controlling Semaphor ...
D. Marshall : Semaphore Operations ...
D. Marshall : Arrays and Strings
D. Marshall : Single and Multi-dim ...
D. Marshall : Functions and Arrays ...
C.Faq : Question 1.31
C.Faq : Question 15.13
C.Faq : Question 16.3
C.Faq : Question 2.6

Figure 3. A user view of the knowledge map and the content of a particular cell in Knowledge Sea portal

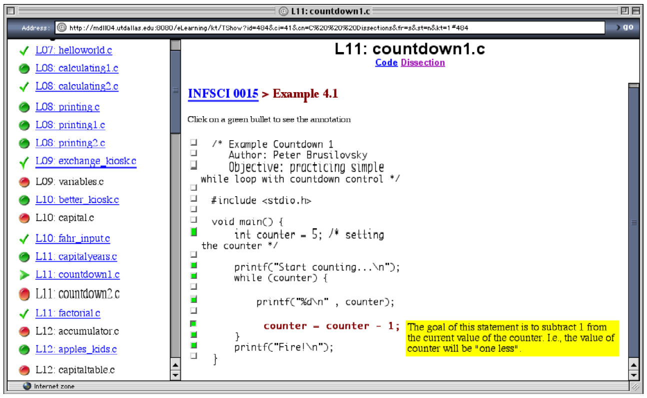
In Knowledge Sea, a student searches for learning material relevant to a particular lecture by examining resources that were classified by the system in the same cell with the lecture notes for this lecture and in the cells around it. The system also supports "horizontal" navigation between the fragments of learning material (Figure 3). Since our architecture supports "chaining" of portals and services, Knowledge Sea is currently used as a value-adding service providing access to content that has not yet been structured by the course author. The current version of Knowledge Sea is adaptive. It queries the student model server about group navigation patterns and provides adaptive navigation support using adaptive annotation. The kind of navigation support provided by Knowledge Sea is known as social navigation [18]. Using changing background and icon colors, it attracts user attention to cells and resources that were most often visited and annotated by the user herself and by a group of users with similar goals and knowledge. Current versions of Knowledge Sea and KnowledgeTree are Java-based and run under a Tomcat server on two different PC-based computers.

3.2 The Activity Servers and the Services

Our main activity servers have been developed specifically for the teaching of programming since it was the main application area of KnowledgeTree. Each server supports authoring of a specific kind of interactive learning activity, stores all authored activities, and maintains the student's interaction with a selected activity of this kind. Since all these servers have been reported elsewhere, we are just providing a brief summary of their capabilities in this paper:

The WebEx system serves interactive annotated program examples [7], the QuizPACK serves parameterized programming questions [33], and WADEIn [11] serves adaptive demonstrations and exercises related to expression evaluation. At the moment, the only fully adaptive server is WADEIn (though QuizPACK questions are personalized).

To provide adaptive access to WebEx examples, we use a value-adding service NavEx. NavEx is also a domain-dependent service focused on teaching programming. NavEx is able to extracts domain concepts from the raw code of programming examples. Using this knowledge extracted from examples and knowledge about individual student obtained from the student model server it is able to recommend some examples as ready-to-be-learned (green bullet on Figure 4) and restrict access to the examples that are currently too complicated (red bullet on Figure 4). NavEx can be accessed through the KnowledgeTree portal. It dramatically helps a teacher to organize access to examples.


L11: countdown1.c

Address: http://mdll04.utdallas.edu:8080/eLearning/kt/TShow?id=484&ci=41&cn=C%20%20%20Dissections&fr=s&st=n&kt=1#484

✓ L07: helloworld.c
● L08: calculating1.c
✓ L08: calculating2.c
● L08: printing.c
● L08: printing1.c
● L08: printing2.c
✓ L09: exchange_kiosk.c
● L09: variables.c
● L10: better_kiosk.c
● L10: capital.c
✓ L10: fahr_input.c
● L11: capitalyears.c
➤ L11: countdown1.c
● L11: countdown2.c
✓ L11: factorial.c
● L12: accumulator.c
● L12: apples_kids.c
● L12: capitaltable.c

L11: countdown1.c
Code Dissection

INFSCI 0015 > Example 4.1

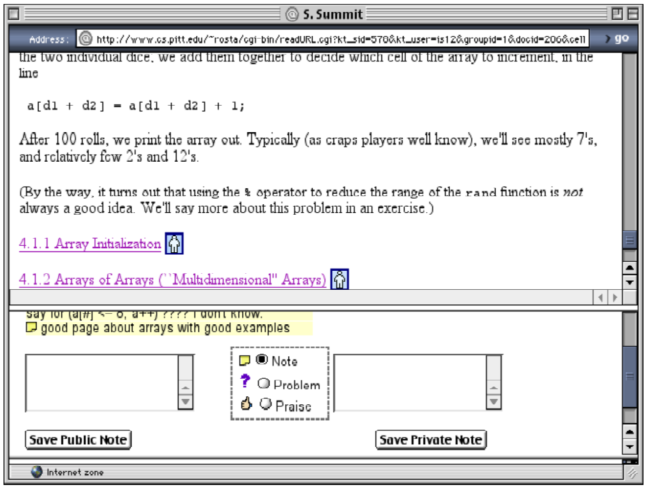
Click on a green bullet to see the annotation

```
  /* Example Countdown 1
     Author: Peter Brusilovsky
     Objective: practicing simple
while loop with countdown control */

  #include <stdio.h>

  void main() {
      int counter = 5; /* setting
the counter */

      printf("Start counting...\n");
      while (counter) {

          printf("%d\n" , counter);

            counter = counter - 1;
      }
      printf("Fire!\n");
  }
```

The goal of this statement is to subtract 1 from the current value of the counter. I.e., the value of counter will be "one less".

Figure 4. Adaptive access to annotated examples through the NavEx service

Another value-adding service AnnotatED is domain independent. AnnotatED provides an ability to annotate static and interactive content. AnnotatED is not conceptually different from other known annotation services; however, it complies with KnowledgeTree authentication and student modeling protocols. Using chaining of services, any piece of content or an interactive activity could be called from any portal through AnnotatED (Figure 5). Annotations created with AnnotatED are reflected in the student model and used to support adaptive social navigation. Currently all content integrated by Knowledge See portal and selected content references by KnowledgeTree portal are accessed through AnnotatED.


S. Summit
Address:
http://www.cs.pitt.edu/~rosta/cgi-bin/readURL.cgi?kt_sid=578&kt_user=is12&groupid=1&docid=206&cell
go
Note
Problem
Praise
Save Public Note
Save Private Note
Internet zone

the two individual dice, we add them together to decide which cell of the array to increment, in the line

```
a[d1 + d2] = a[d1 + d2] + 1;
```

After 100 rolls, we print the array out. Typically (as craps players well know), we'll see mostly 7's, and relatively few 2's and 12's.

(By the way, it turns out that using the `%` operator to reduce the range of the `rand` function is *not* always a good idea. We'll say more about this problem in an exercise.)

4.1.1 Array Initialization

4.1.2 Arrays of Arrays (``Multidimensional'' Arrays)

say for (a[#] <= 6, a++) ???? I don't know.

good page about arrays with good examples

Figure 5. A view of an online C tutorial page through the annotation service AnnotatED

Technically, all activity servers and services can be considered self-containing Web server-side applications. They were developed using different technologies and run on different platforms. WebEx and NavEx are Java based and run under Tomcat from different computers. WADEIn is based on a Java applet, QuizPACK is a classic CGI program developed in C++ and running on a Sun server. AnnotatED is a Perl-based application delivered from a Unix server. Quite often a student uses the capabilities of all these services in one session without ever realizing that this session was supported by several applications running on different geographically distributed computers. All these servers implement one simple, transparent login protocol, a resource delivery protocol, and a student modeling protocol. They can work with any portal and student-modeling server supporting these protocols. The ability to make the current "zoo" of resources work seamlessly is a good argument for the use of our architecture.

3.3 The Student Model Server CUMULATE

The CUMULATE Student Model Server is a new implementation of the event-based centralized student modeling architecture that we have investigated in the past [5]. The overall idea of this server is to collect evidence (events) about student learning from multiple servers that interact with the student, then process these events into a shareable set of student parameters that are typical for classic student models (Figure 6). Each event specifies the activity server, the kind of event (i.e., page is read, question is answered, example is analyzed), the activity producing the event and an outcome of the event. The collected time-stamped events are stored in the *event storage* part of the student model that is implemented as a relational database (Figure 7). External and internal *inference agents* process the flow of events in different ways and update the values in the *inference model* part of the server, in the form of pairs (property-value) and triples (property-object-value). These pairs and triples form the upper level of the model which can then be queried by activity servers and portals using a standard querying protocol.

Each inference agent is responsible for maintaining a specific property in the inference model. For example, one agent can process the sequence of events for the purpose of deducing the current motivation level of the student while another agent may process it to infer the student's current level of knowledge for each of the course topics. One usually expects that most inference agents will reside on the student model server; however, CUMULATE allows adding external inference agents that can run on external servers.

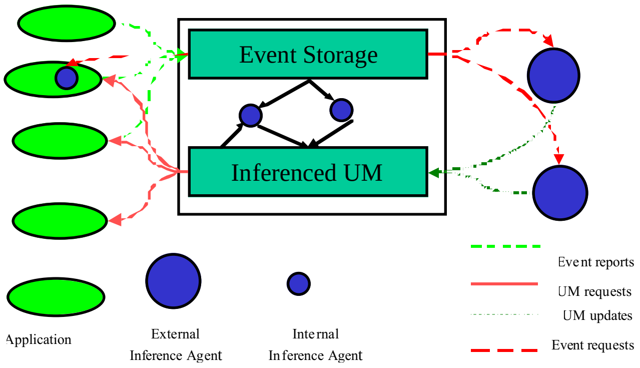
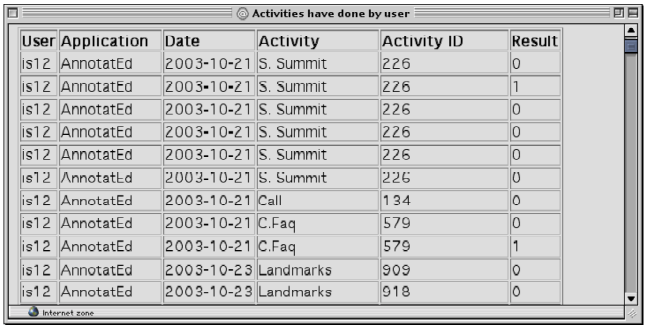


Figure 6. Centralized student modeling in CUMULATE

Currently CUMULATE supports two levels of queries to the student model. A value query specifies a user, a property and, if appropriate a list of objects (for example, documents, or course topics). It returns a value of the property or a list of values for requested objects. An evidence query is restricted to a single property and a single object, but in addition to the value it returns a structured summary of all events that the agent has used in calculating the final value. It is important that CUMULATE supports both student and group modeling, i.e., inference agents, can appropriately produce property values for either single users or the whole group of users.



User	Application	Date	Activity	Activity ID	Result
is12	AnnotatEd	2003-10-21	S. Summit	226	0
is12	AnnotatEd	2003-10-21	S. Summit	226	1
is12	AnnotatEd	2003-10-21	S. Summit	226	0
is12	AnnotatEd	2003-10-21	S. Summit	226	0
is12	AnnotatEd	2003-10-21	S. Summit	226	0
is12	AnnotatEd	2003-10-21	S. Summit	226	0
is12	AnnotatEd	2003-10-21	Call	134	0
is12	AnnotatEd	2003-10-21	C.Faq	579	0
is12	AnnotatEd	2003-10-21	C.Faq	579	1
is12	AnnotatEd	2003-10-23	Landmarks	909	0
is12	AnnotatEd	2003-10-23	Landmarks	918	0

Figure 7. A fragment of event protocol viewed through one of CUMULATE tools

The current version of this architecture uses two essentially different inference agents. A knowledge-inferring agent processes events by user and topic. It extracts records of different learning activities related to this topic and attempts to deduce the current student knowledge level of this topic. Student knowledge levels are currently used by WADEIn and the version of QuizPACK under development. A simple activity-inferring agent processes events by activity. It extracts all events related to a specific learning activity, counts the number of visits and annotation for this activity for any user and group, and calculates activity levels. Activity levels are currently used by KnowledgeSea and AnnotatED to support social navigation. In addition to these agents, CUMULATE provides a number of maintenance tools for a server administrator as well as several tools for teachers and students to examine the content of the student models (Figure 7).

3.4 The Protocols

The current version of KnowledgeTree implements a lightweight version of all the communication protocols that were necessary to make this version function in a classroom. When implementing these protocols, our group neither intended to "make it right" from the first attempt nor wanted to set up standards for other similar projects to adhere. Our main goal was to explore a distributed architecture in a practical context to find out which protocols are necessary for making the whole architecture flexible and what kind of information should be passed between servers of different kind. As for the running implementation, we were mostly concerned with simplicity and efficiency. Establishing standards for the communication protocols is certainly necessary to make the architecture widespread; however, we consider our current research a necessary precondition for this process. As soon as the needs and problems have been sufficiently explored, the standards themselves should be crafted by a group of like-minded stakeholders who have experience with distributed E-Learning architectures, understand different sides of the problem and are familiar with existing protocols and standardization efforts.

The first of the protocols we had to develop was the transparent authentication protocol. Each portal or activity server should be able to recognize individual users to report events to the student model and, possibly, to adapt to the user. The authentication should happen transparently without the need for the user to log in to each of the multiple servers used during one session.

Transparent authentication (or single sign-on) is a recognized industrial problem. A number of solutions were created over the last few years including Microsoft Passport (http://www.passport.net/) and SAML [14]. We use a simpler approach that we have inherited from our earlier ELM-ART [43] and InterBook [8] systems: all authentication data are passed from server to server as part of the launch URL (course URLs for a portal; activity URLs for an activity server). Currently a user ID, a group ID and a session ID are passed as a part of the http GET request. The session ID is generated for every student at the beginning of every session. A session ID - user ID pair can be used by any server to check the validity of each user. Every event or request sent to the student model should have a valid session ID in order to be processed. This authentication approach is very efficient, provides sufficient (for our context) protection and allows "chaining" of services (i.e., not only can a portal call a service, but also one portal can call another portal or one service can call another).

A protocol for communication between a portal and a pool of resources is an active research issue in the field of E-Learning. Ideally, this protocol should allow querying a pool for relevant content and services, to find properties of an activity or service, and to launch it. In addition to these protocols, a distributed E-Learning system should have a resource discovery framework. Currently there are two approaches to address the discovery issue: a centralized broker-based approach [24; 38] and peer-to-peer [31; 39]. In particular, EDUTELLA [31] suggest a very impressive architecture that can resolve all listed needs.

Our approach is conceptually similar to EDUTELLA. Our services and activities are identified by a unique locator (URL), but we use a very restricted set of metadata (learning outcomes, prerequisites, and complexity) and our exchange protocols are much simpler. The resource discovery issue has not been addressed in the current version of KnowledgeTree. Currently, we simply "tell" the portal about all existing activity servers.

The semantics of the student modeling protocols has already been presented in the previous section. Here we only wish to add that it is the least investigated problem in the field of E-Learning. As a result, a significant fraction of our work on KnowledgeTree has been devoted to investigating the problems of student modeling for E-Learning.

Technically, all our inter-server communication protocols are based on a simple model: HTTP GET request - XML reply. Even the student modeling event messages are sent by all servers to the student modeling server in the form of GET requests. The use of structured GET requests to pass information was inherited from our earlier work on distributed E-Learning [12] and can be currently considered rather archaic. We have analyzed a number of standard approaches, including RPC, CORBA, and SOAP and have very seriously considered XML-RPC, which was used by another distributed system, ActiveMath [29]. However, at the current state of our research we decided to stay with our current approach since it satisfies our research needs completely while being dramatically faster than other protocols. The latter issue is critical for any classroom experiments with a distributed architecture where frequent inter-server communication should not be allowed to slow down the student interface. The focus of our work is to determine what kind of information has to be communicated, not to find the best communication protocol. Since the protocol details are hidden from developers of portals and services (they use it through information-focused API), we can seamlessly adopt any commonly accepted efficient carrier protocol.

4. SIMILAR WORK

This section attempts to summarize similar and complementary research and development efforts. A number of references and comparisons were provided in the main body of the paper. Here we provide a systematic review, grouping similar works into three clusters: reusability standards, communication frameworks and research level architectures.

4.1 Reusability Standards

A number of educational technology standards are currently supporting content reusability goals. Several standardization bodies such as AICC (http://www.aicc.org), IEEE LTSC (http://ltsc.ieee.org), ADL (http://www.adlnet.org/), and IMS (http://www.imsproject.org/) have issued a number of standards and drafts focused on reusability. From the prospects of our project, all standards can be split into two groups - *information exchange* standards and *interoperability* standards. Information exchange standards prescribe the way to store information about various entities of E-learning, from learning objects and packages to learners themselves. If standardized, this information can be easily moved from system to system, supporting the separation of learning contents and learners from LMS. Information exchange standards have less relevance to our architecture. First of all, these standards emphasize data storage, while our architecture represents the communication viewpoint. As long as a portal or a value-adding service can receive requested information about activities from activity servers, it does not matter how this information is represented or stored. Secondly, current standards were established before the needs being explored in KnowledgeTree were properly understood. As a result, the information that a standards-based system stores about content and users is almost useless for the adaptation needs of KnowledgeTree, while the information that is vital for adaptation is not found. To deal with this problem other research teams focused on personalizing E-Learning, attempted to combine several standards while complementing them with additional information [19].

In contrast, *interoperability* standards, which ensure that different components of E-Learning systems can work together, are very relevant to KnowledgeTree. Of all the interoperability standards, the one most similar to KnowledgeTree is the so-called CMI standard, which was originally introduced by AICC [3] and later adopted by IEEE LTSC and ADLI as a part of SCORM [1; 2]. CMI anticipates a very advanced level of communication between an LMS and a fragment of learning content. A content object can store and query information about student performance related to multiple educational objectives in an LMS. This is similar to the classic overlay student modeling in intelligent tutoring systems, which is capable to support adaptation. In addition, a course creator can associate advanced sequencing rules with a structured set of objects allowing it to inherently bear a number of adaptation effects within these sequenced learning objects. The most recent version of SCORM [2] can be considered several steps beyond the monolithic LMS of today in most aspects, including adaptivity. Yet, when one of the research groups working on service-based E-Learning did an honest attempt to use CMI for student modeling, it discovered conceptual and technical problems [15].

SCORM successfully separates learning content from its LMS allowing the content to be used with multiple LMSs. However, it has failed to separate learning content from sequencing and student modeling. SCORM recognizes only two components in a distributed E-Learning architecture - reusable content and the LMS. In SCORM, student modeling is "hardwired" into fragments of "intelligent" content. As a result, the reusability of adaptive content dramatically decreases since it becomes "tuned" to a specific student modeling approach and a specific set of objectives. In contrast, KnowledgeTree and similar advanced architectures use a student model server to separate the student modeling from reusable educational activities. Different servers can support different student modeling approaches and different domain concepts for the same activities. It supports a greater flexibility and makes these activities highly reusable.

Content sequencing in SCORM is also hardwired into the structured content. It immediately excludes adaptive use of external content (open corpus) which conflicts with adaptive portals and value-adding services. In contrast, KnowledgeTree separates content from adaptive sequencing leaving the duty of sequencing to portals and value-adding content integration services. It allows the use of open corpus content and chaining of value-adding services and portals.

Despite of many advanced features introduced in of SCORM, its support of personalized E-Learning falls behind the state-of-the-art level in the field of adaptive educational systems. We think that the future student model servers should be based not on these standards, but on the earlier work on user model systems and servers, a popular research topic within the user modeling area [27]. Both our group and the University of Sydney group that developed the first comprehensive student modeling servers CUMULATE and Personis [26] have benefited from solid past experience developing user modeling architectures [5; 25].

4.2 Communication Frameworks

Communication frameworks such as OKI (http://web.mit.edu/oki/) or uPortal (http://www.uportal.org/) champion the ideas of component-based, distributed E-Learning.

The focus of uPortal project [23] is a seamless presentation of information coming from multiple external services (known as *channels*) through the user interface of an educational portal. KnowledgeTree and uPortal share the same portal/service architecture, but focus on different aspects of it. KnowledgeTree focuses on content while uPortal excels in presentation. As a result, these frameworks complement each other. As a research framework, KnowledgeTree has the luxury of ignoring the presentation aspect because it launches external services and activities in separate browser windows and frames. However, a practical system developed on the basis of KnowledgeTree will certainly benefit from uPortal's approach for assembling a coherent presentation.

OKI architecture [40] and KnowledgeTree architecture have a lot in common. Both frameworks define the generic architecture of an E-Learning system as being based on components and both focus on the communication interfaces between the components. As a result, components become highly reusable and replaceable. Unlike storage-oriented standards that prescribe what should be inside components, both KnowledgeTree and OKI standardize communication interfaces and ignore the internal organization of the components. Yet KnowledgeTree and OKI are quite different because they focus on different sides of the E-Learning process. KnowledgeTree focuses on the educational side of E-Learning, representing the educational needs of students and teachers as well as the needs of service and content providers. It originates from the domain's adaptive educational systems, strives to support an advanced educational process and mostly ignores the needs of university administration. In contrast, OKI focuses mostly on the administrative and management side of E-Learning, representing the needs of university administration and class management needs of teachers. It originates from campus administration systems and suggests a finer-grain component-based architecture. As a result KnowledgeTree and OKI complement each other.

4.3 Research Level Architectures

The problems of developing distributed adaptive and intelligent educational systems based on shared educational resources have been originally explored in the field of ITS [12; 22; 30; 35; 36]. At that time, the lack of matching work in other fields and appropriately advanced technology in general limited these pioneer works to the theoretical level, with a few simple lab systems. The Web as a unified platform for E-Learning has changed the situation dramatically. The race for E-Commerce, E-Learning, enterprise systems, Web services, and personalization, has brought to life many technologies that can now be used for the development of adaptive, distributed E-learning and has inspired a number of research streams.

The focus of adaptive distributed learning research has now moved to the Adaptive Hypermedia and E-Learning research communities. Among several projects that focus on adaptive E-Learning, two projects are most close to the KnowledgeTree vision of distributed personalized service-based E-Learning.

ELENA, an international project focuses on personalization in distributed E-Learning Networks [19; 20; 21]. ELENA's architecture has several similarities with KnowledgeTree. It recognizes such entities as resource providers and value-adding services, though it currently integrates a student portal with the student model server in a personal learning assistant peer. As a result, it has no specific architectural component to support the needs of a teacher. An important difference between the projects is that ELENA starts from the analysis of global needs and focuses more on interoperability and technology. To support interoperability, ELENA attempts to define precisely the format for stored and communicated knowledge. KnowledgeTree starts with the needs of humans - the main stakeholders of the E-Learning process and focuses on the content rather than format. These projects are quite complementary and we hope, that we will be able to integrate our prospects in the context of ProLearn Network of Excellence.

Another relevant project originates from Ireland [15; 16; 17]. The group in Trinity College Dublin investigates value-adding services for the adaptive presentation of reusable content. This work is similar to the KnowledgeTree architecture in two main aspects. First, it stresses the need to move from reusable objects to adaptive reusable services. Secondly, through the mechanism of narration, it attempts to support the teacher’s need to be an active integrator of educational content. This is exactly the same need that is anticipated and supported by a KnowledgeTree portal.

5. A SUMMARY OF EXPERIENCE

As we have mentioned, KnowledgeTree has been extensively used in teaching several real courses for about two years. Over this period we have created three different course trees with KnowledgeTree and two different knowledge maps with Knowledge Sea. Three main activity servers are currently hosting more than 200 interactive activities. In addition to that, Knowledge Sea provides access to more than a 1000 static C-tutorial pages that are currently served through AnnotatED. This large and comprehensive volume of learning material makes it very easy to structure a new course focused on teaching C programming. In addition, it allows us to use KnowledgeTree as a primary tool in supporting practical courses. We think that these are good proofs of the KnowledgeTree concept.

We have never run a subjective evaluation of KnowledgeTree as architecture; however we have evaluated several components, including Knowledge Sea [13] and several activity servers. All these evaluation brought very positive and even remarkable results. Since all subjective evaluation questionnaires included a free-form feedback, we have collected a good amount of unsolicited praise for the system in general. The students have most of all appreciated the variety of tools provided and the simplicity of use of these tools, through a single-entry portal.

The presence of the CUMULATE server, which records time-stamped student performance with every activity made it very easy to observe what the students were doing with the system. Most interesting for us was to discover that the profiles of the activity usage differ a lot from user to user. Some users generated several hundred records with one or two of our tools while nearly ignoring the rest. We hypothesis that different kinds of activities correspond better to different learning styles. It provides another reason to choose the KnowledgeTree approach for providing access to very diverse activities from a single portal. Overall, the students used KnowledgeTree and its components a great deal. For merely a single QuizPACK server, the average number of questions answered by a user over the duration of the course was more than 180, while some students attempted more than 500 (!) questions. The results of user ranking of their learning tools showed that the activity servers offering advanced interactive activities were more popular than the lecture notes and textbooks that are the traditional "static" learning tools in a university.

6. CONCLUSION

This paper proposes an architecture for adaptive E-Learning based on distributed reusable intelligent learning activities that integrate the benefits provided by modern LMS and educational material repositories with the power of ITS and AH technologies. This architecture addresses both the component-based development of adaptive systems and the teacher-level reusability. We have started by implementing the core functionality of the system within our local group by using some rather simple approaches to implement the required protocols.

Some other groups driven by similar goals have proposed other architectures that match our vision [16; 19; 26; 29]. A significant amount of work and cooperation between several research groups will be required to turn the proposed architectures into the common practice of E-Learning. Fortunately, our work shares many goals with several other active Web-related research areas, enabling us to re-use possible standards, solutions and ideas from these areas. It gives our group, along with other similarly-motivated groups, a good chance of succeeding in bringing this new generation of adaptive E-Learning systems and tools to the educational world.

7. REFERENCES

[1] ADLI Sharable Content Object Reference Model (SCORM), Advanced Distributed Learning Initiative, 2003, available online at http://www.adlnet.org/

[2] ADLI Overview of SCORM 2004, Advanced Distributed Learning Initiative, 2004, available online at http://www.adlnet.org/screens/shares/dsp_displayfile.cfm?fileid=992

[3] AICC AGR010 - Web-based Computer Managed Instruction (CMI). Ver. 1.0, Aviation Industry CBT

Committee, 1998, available online at http://www.aicc.org/pages/down-docs-index.htm.

[4] Blackboard Inc. Blackboard Course Management System, Blackboard Inc., 2002, available online at http://www.blackboard.com/

[5] Brusilovsky, P. Student model centered architecture for intelligent learning environment. In: Proc. of Fourth International Conference on User Modeling, (Hyannis, MA, 15-19 August 1994), MITRE, 31-36, available online at http://www2.sis.pitt.edu/~peterb/papers/UM94.html.

[6] Brusilovsky, P. Adaptive and Intelligent Technologies for Web-based Education. Künstliche Intelligenz, , 4 (1999), 19-25, available online at http://www2.sis.pitt.edu/~peterb/papers/KI-review.html.

[7] Brusilovsky, P. WebEx: Learning from examples in a programming course. In: Fowler, W. and Hasebrook, J. (eds.) Proc. of WebNet'2001, World Conference of the WWW and Internet, (Orlando, FL, October 23-27, 2001), AACE, 124-129.

[8] Brusilovsky, P., Eklund, J., and Schwarz, E. Web-based education for all: A tool for developing adaptive courseware. Computer Networks and ISDN Systems. 30, 1-7 (1998), 291-300.

[9] Brusilovsky, P. and Maybury, M.T. From adaptive hypermedia to adaptive Web. Communications of the ACM, 45, 5 (2002), 31-33.

[10] Brusilovsky, P. and Miller, P. Course Delivery Systems for the Virtual University. In: Tschang, T. and Della Senta, T. (eds.): Access to Knowledge: New Information Technologies and the Emergence of the Virtual University. Elsevier Science, Amsterdam, 2001, 167-206.

[11] Brusilovsky, P. and Nijhawan, H. A Framework for Adaptive E-Learning Based on Distributed Re-usable Learning Activities. In: Driscoll, M. and Reeves, T.C. (eds.) Proc. of World Conference on E-Learning, E-Learn 2002, (Montreal, Canada, October 15-19, 2002), AACE, 154-161.

[12] Brusilovsky, P., Ritter, S., and Schwarz, E. Distributed intelligent tutoring on the Web. In: du Boulay, B. and Mizoguchi, R. (eds.) Artificial Intelligence in Education: Knowledge and Media in Learning Systems. IOS, Amsterdam, 1997, 482-489.

[13] Brusilovsky, P. and Rizzo, R. Using maps and landmarks for navigation between closed and open corpus hyperspace in Web-based education. The New Review of Hypermedia and Multimedia, 9 (2002), 59-82.

[14] Byous, J. Single sign-on simplicity with SAM: An overview of single sign-on capabilities based on the Security Assertions Markup Language (SAML) specification, Sun Microsystems, Inc., 2002, available online at http://java.sun.com/features/2002/05/single-signon.html.

[15] Conlan, O., Dagger, D., and Wade, V. Towards a standards-based approach to e-Learning personalization using reusable learning objects. In: Driscoll, M. and Reeves, T.C. (eds.) Proc. of World Conference on E-Learning, E-Learn 2002, (Montreal, Canada, October 15-19, 2002), AACE, 210-217.

[16] Conlan, O., Wade, V., Gargan, M., Hockemeyer, C., and Albert, D. An architecture for integrating adaptive hypermedia services with open learning environments. In: Barker, P. and Rebelsky, S. (eds.) Proc. of ED-MEDIA'2002 - World Conference on Educational Multimedia, Hypermedia and Telecommunications, (Denver, CO, June 24-29, 2002), AACE, 344-350.

[17] Dagger, D., Conlan, O., and Wade, V.P. An architecture for candidacy in adaptive eLearning systems to facilitate the reuse of learning Resources. In: Rossett, A. (ed.) Proc. of World Conference on E-Learning, E-Learn 2003, (Phoenix, AZ, USA, November 7-11, 2003), AACE, 49-56.

[18] Dieberger, A., Dourish, P., Höök, K., Resnick, P., and Wexelblat, A. Social Navigation: Techniques for Building More Usable Systems. interactions, 7, 6 (2000), 36-45.

[19] Dolog, P., Gavriloaie, R., Nejdl, W., and Brase, J. Integrating Adaptive Hypermedia Techniques and Open RDF-Based Environments. In: Proc. of The Twelfth International World Wide Web Conference, WWW 2003, (Budapest, Hungary, 20-24 May, 2003), 88-98.

[20] Dolog, P. and Henze, N. Personalization Services for Adaptive Educational Hypermedia. In: Proc. of International Workshop on Adaptivity and User Modelling in Interactive Systems (ABIS'2003), (Karlsruhe, Germany, October 2003).

[21] Dolog, P. and Nejdl, W. Challenges and benefits of the semantic Web for user modeling. In: De Bra, P., Davis, H., Kay, J. and schraefel, m. (eds.) Proc. of AH2003: Workshop on adaptive hypermedia and adaptive Web-based systems, (Budapest, Hungary, Eindhoven University of Technology, 99-111.

[22] Eliot, C., Woolf, B., and Lesser, V. Knowledge extraction for educational planning. In: Proc. of Workshop on Multi-Agent Architectures for Distributed Learning Environments at AIED'2001, (San Antonio, May 19, 2001), available online at http://julita.usask.ca/mable/eliot.doc.

[23] JA-SIG A general overview of uPortal 2.x, Java Architectures SIG, 2003, available online at http://mis105.mis.udel.edu/ja-sig/uportal/architecture/uPortal_architecture_overview.pdf.

[24] Karagiannidis, C., Sampson, D.G., and Cardinali, F. An architecture for Web-based e-Learning promoting re-usable adaptive educational e-content. Educational Technology & Society, 5, 4 (2002).

[25] Kay, J. The UM toolkit for cooperative user models. User Modeling and User-Adapted Interaction, 4, 3 (1995), 149-196.

[26] Kay, J., Kummerfeld, B., and Lauder, P. Personis: A server for user modeling. In: De Bra, P., Brusilovsky, P. and Conejo, R. (eds.) Proc. of Second International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'2002), (Málaga, Spain, May 29-31, 2002), 201-212.

[27] Kobsa, A. Generic user modeling systems. User Modeling and User Adapted Interaction, 11, 1-2 (2001), 49-63, available online at http://www.ics.uci.edu/~kobsa/papers/2001-UMUAI-kobsa.pdf.

[28] Machado, I., Martins, A., and Paiva, A. One for all and all for one: a learner modelling server in a multi-agent

platform. In: Kay, J. (ed.) SpringerWienNewYork, Wien, 1999, 211-221.

[29] Melis, E., Andrès, E., Büdenbender, J., Frishauf, A., Goguadse, G., Libbrecht, P., Pollet, M., and Ullrich, C. ActiveMath: A web-based learning environment. International Journal of Artificial Intelligence in Education, 12, 4 (2001), 385-407.

[30] Murray, T. A Model for Distributed Curriculum on the World Wide Web. Journal of Interactive Media in Education, (1998), available online at http://www-jime.open.ac.uk/98/5/.

[31] Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmér, M., and Risch, T. EDUTELLA: A P2P Networking Infrastructure Based on RDF. In: Proc. of 11th International World Wide Web Conference, (Honolulu, Hawaii, USA, 7-11 May 2002), available online at http://www2002.org/CDROM/refereed/597/index.html.

[32] Oda, T., Satoh, H., and Watanabe, S. Searching deadlocked Web learners by measuring similarity of learning activities. In: Proc. of Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98), (San Antonio, TX, August 16-19, 1998), available online at http://www.sw.cas.uec.ac.jp/~watanabe/conference/its98workshop1.ps.

[33] Pathak, S. and Brusilovsky, P. Assessing Student Programming Knowledge with Web-based Dynamic Parameterized Quizzes. In: Barker, P. and Rebelsky, S. (eds.) Proc. of ED-MEDIA'2002 - World Conference on Educational Multimedia, Hypermedia and Telecommunications, (Denver, CO, June 24-29, 2002), AACE, 1548-1553.

[34] Rios, A., Millán, E., Trella, M., Pérez, J.L., and Conejo, R. Internet based evaluation system. In: Lajoie, S.P. and Vivet, M. (eds.) Artificial Intelligence in Education: Open Learning Environments. IOS Press, Amsterdam, 1999, 387-394.

[35] Ritter, S., Brusilovsky, P., and Medvedeva, O. Creating more versatile intelligent learning environments with a component-based architecture. In: Goettl, B.P., Halff, H.M., Redfield, C.L. and Shute, V.J. (eds.) Lecture Notes in Computer Science, Vol. 1452. Springer Verlag, Berlin, 1998, 554-563.

[36] Ritter, S. and Koedinger, K.R. An architecture for plug-in tutor agents. Journal of Artificial Intelligence in Education, 7, 3/4 (1996), 315-347.

[37] Soller, A. and Lesgold, A. A computational approach to analysing online knowledge sharing interaction. In: Hoppe, U., Verdejo, F. and Kay, J. (eds.) Artificial intelligence in education: Shaping the future of learning through intelligent technologies. IOS Press, Amsterdam, 2003, 253-260.

[38] Specht, M., Kravcik, M., Pesin, L., and Klemke, R. Learner's Lounge: Information Brokering for the Adaptive Learning Environment. In: Driscoll, M. and Reeves, T.C. (eds.) Proc. of World Conference on E-Learning, E-Learn 2002, (Montreal, Canada, October 15-19, 2002), AACE, 2510-2512.

[39] Ternier, S., Duval, E., and Vandepitte, P. LOMster: Peer-to-peer Learning Object Metadata. In: Barker, P. and Rebelsky, S. (eds.) Proc. of ED-MEDIA'2002 - World Conference on Educational Multimedia, Hypermedia and Telecommunications, (Denver, CO, June 24-29, 2002), AACE, 1942-1943.

[40] Thorne, S., Shubert, C., and Merriman, J. OKI Architectural Overview, Open Knowledge Initiative, 2002, available online at http://web.mit.edu/oki/learn/whtpapers/ArchitecturalOverview.pdf.

[41] Verhoeven, B., Cardinaels, K., Van Durm, R., Duval, E., and Olivié, H. Experiences with the ARIADNE pedagogical document repository. In: Proc. of ED-MEDIA'2001 - World Conference on Educational Multimedia, Hypermedia and Telecommunications, (Tampere, Finland, June 25-30, 2001), AACE, 1949-1954.

[42] WebCT WebCT Course Management System, Lynnfield, MA, WebCT, Inc., 2002, available online at http://www.webct.com

[43] Weber, G. and Brusilovsky, P. ELM-ART: An adaptive versatile system for Web-based instruction. International Journal of Artificial Intelligence in Education, 12, 4 (2001), 351-384, available online at http://cbl.leeds.ac.uk/ijaied/abstracts/Vol_12/weber.html.

[44] Weber, G., Kuhl, H.-C., and Weibelzahl, S. Developing adaptive internet based courses with the authoring system NetCoach. In: Bra, P.D., Brusilovsky, P. and Kobsa, A. (eds.) Proc. of Third workshop on Adaptive Hypertext and Hypermedia, (Sonthofen, Germany, July 14, 2001), Technical University Eindhoven, 35-48, available online at http://wwwis.win.tue.nl/ah2001/papers/GWeber-UM01.pdf.