SKILL - a Scalable Internet-Based Teaching and Learning System*

Gustaf Neumann, Jana Zirvas
{neumann,zirvas}@nestroy.wi-inf.uni-essen.de
Information Systems and Software Techniques
University of Essen, Germany

Abstract

Abstract: This paper describes the architecture of a scalable Internet-based teaching and learning system (SKILL) currently developed at the University of Essen, Germany. The system will be developed first with the focus to the department Information Systems and Software Technologies, but is designed as a scalable architecture for a much wider scope. The primary objective of SKILL is to provide students a collaborative and adaptive learning environment utilising new web technologies proposed by W3C. Basic components of SKILL are (a) course material based on concepts which are organised in an ordinal rating derived from pre-requirements (b) annotation facility suited for collaboration work and (c) configuration environment for tailoring the system.

1 Introduction

The curricula of many or maybe most German universities are currently under discussion. On the one hand public universities have to face a growing competition from private universities, on the other hand there is a strong desire from government and industries to shorten the length of the studies and to allow higher flexibility and more specialisations. As a consequence classes become more heterogeneous. Traditional ways of teaching are not longer sufficient to cover the continuous' growing requirements on students. In our project we are developing a scalable Internet-based teaching and learning system. The main goal of the system is to cope with the different knowledge level of the students and with their different learning preferences. The new learning environment should be more flexible in presenting contents by adapting individual needs and should allow collaborative work by supporting learning groups. In our architecture the students can extend the system in various ways, these extensions are the source for improved revisions of the courses.

The system is currently in the design phase. In this paper we will discuss our design of the architecture and we will address various implementation considerations.

2 SKILL Functionality

Basis for the SKILL system is the actual learning material developed at the department which is currently available on the Web in form of course slides converted to HTML, additional references, and handouts. Starting from this basic pool of concepts a knowledge pool is generated. The knowledge pool contains the material of all courses offered by the department. The courses elements are course

*Published in: Proceedings of WebNet 98, World Conference on WWW, Internet and Intranet, AACE, Orlando, Fl, November, 1998.

units and concepts. A course is composed of course units. Every unit contains concepts, which require some prior knowledge (concepts as well). Every concept is represented by one or more documents and additional navigation information. A document is either a content-document or an exercise-document.

When a course is designed, a set of relevant concepts is selected by a teacher from the knowledge pool (see Figure 1). The selected concepts are grouped into course units and a navigation path through

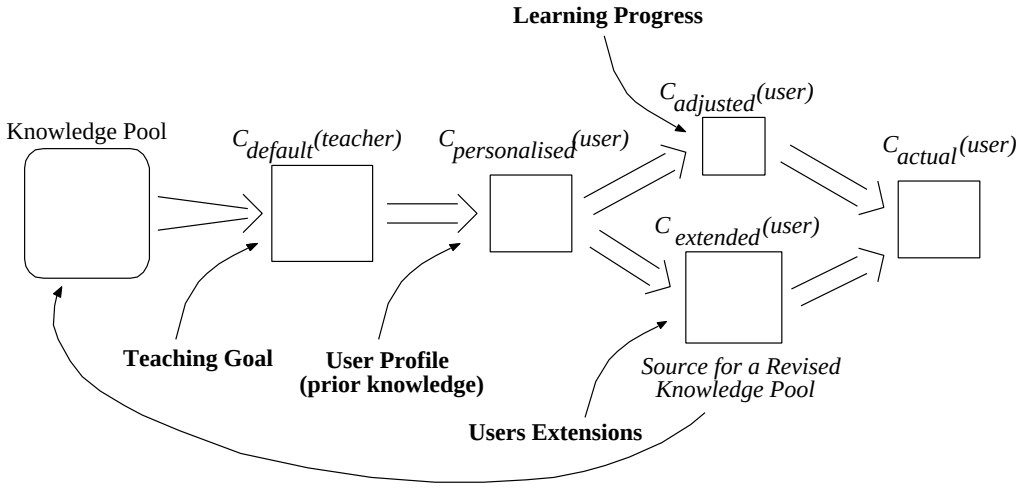

Figure 1: Course Refinement Levels

these concepts is defined. Every concept can be part of multiple courses. The navigation path forms a frame for the course and will be held physically separated from course contents. This separation between course contents and the navigation path can be kept on the logical level or can be resolved by integrating the navigation path into a visited document.

A course C generated in this way will be called in the sequel "default course" and represents the required knowledge taught by the course as defined by the teacher.

$$C_{default}(teacher) \quad = \quad Knowledge\ Pool + teaching\ goal(teacher) \tag{1}$$

Beside the walk through the default navigation path students can drill down into each concept. Hence for every concept related materials are available as cross references to the actual concept. When a student follows a cross reference he/she leaves the default navigation path, but can resume at this point at some later time.

The users of the SKILL system can be divided into different groups. The individuals working with the system are classified either as "students", "teachers" or "administrators". The union of these classes builds the group "users". Students who are enrolled in the same course can create a new group for collaborative work. This group will be a subgroup of students.

2.1 Adaptivity and Progress Control

Traditional teaching approaches do not consider the heterogenous knowledge level of the students in a class[1]. The SKILL system is designed to allow to customise the students' learning environment according to their personal knowledge and their special interests. In particular a student can choose the course to enrol and specifies a start configuration in the following way: For every course there is a default configuration prepared by the teacher. If a student has already knowledge of some of these concepts obtained by the system, these concepts are marked in the users profile and can be automatically removed. If the student has this information from other sources he can mark the concerned concepts as

[1] From our experience this problem is especially apparent in the area of information systems.

known and is prompted by the control questions from the tutoring component. When these exercises are successfully solved the known concepts are removed (hidden) from the course. If the student does not want to answer the control questions, the concepts to be removed are marked specially until the student solves the exercises at later time. By these customisations a course is personalised by the user.

$$C_{personalised}(user) \quad = \quad C_{default}(teacher) + profile(user) \tag{2}$$

The personalised course is the basis for the students' work during the course. By the continuous learning progress of the student more and more concepts of the course are reached. For all of these concepts, control questions are asked, and more and more concepts are marked as known in the users' profile. This information is kept as well in the system logs and influences the dynamically generated course. By taking the learning progress of a student into account we obtain the adjusted course:

$$C_{adjusted}(user) \quad = \quad C_{personalized}(user) + progress(user) \tag{3}$$

2.2 Collaborativity

In addition to the adaptivity described in the previous section, every student (or learning group) can extend the default information space by *annotations* and additional concepts or single documents (*course extensions*) and share this added information. The system allows extensions for the following purposes:

1. Private annotations
2. Private course extensions
3. Shared annotations and shared course extensions

A student can generate personalised documents by adding private annotations to every available document. A student can add as well new documents (or concepts) to the course in order to associate the collected material with own contents or to develop a proposal for an extended version of the course. A teacher can make the same private extensions on his or her side. All course extensions are added to the navigation structure.

Annotations Every annotation extends a source document – or more precisely – an HTML/XML element of the document (D_e). Two facts distinguish the SKILL annotations architecture from other systems like, e.g., Hyper-G [7]: (1) An annotation is not only associated with the whole document but with a certain part in the document and (2) annotations are related to user groups or individual users and are controlled through an access-control system. By this means a student can add private, public, or only for certain groups accessible annotations which are treated by the system as different annotation sets. Any personalised document is synthesised from the source document and in-line positioned annotations. Valid to a special user are all annotations made by himself as well as all group-related annotations made by members of groups the user belongs to. To differentiate between the source documents content and annotations the latter will be indicated by a special text colour or format which can be specified through CSS. The personalised document can be described as follows:

$$\begin{aligned} D_{personalized}(user) \quad = \quad & D_{source} \\ & + \bigcup_{i=1}^{n} Annotation_i(D_e, user) \\ & + \bigcup_{j=1}^{m} \bigcup_{i=1}^{n} Annotation_i(D_e, group(user)_j) \end{aligned} \tag{4}$$

Course extensions The user (student or teacher) can extend the courses scope as well. As mentioned above a course is built from concepts consisting of a set of documents and a navigation path through these concepts which is held separately. The separation between navigation path and learning contents allows students to extend the course contents without changing source documents. Students or teachers can integrate new documents related to a courses' element C_e content in the navigation path at any position. An extension can be a complete new concept consisting of a set of documents in a course unit or a single document as part of a concept. An extension may contain arbitrary objects such as software packages, seminar papers, etc. All extensions of the course material are subject to access control as well. By adding annotations and further documents learning groups can create their own workspace for collective use.

The addition of new documents can be performed by both students and teachers. Consequently it must be differentiated between authorised additons from teachers with a binding character and addions from students with informational character. This difference can be expressed by membership in different groups which will affect the appearance of an extension in a courses' navigation path.

$$\begin{aligned} C_{extended}(user) &= C_{personalised}(user) \\ &\quad + \bigcup_{i=1}^{n} Extension_i(C_e, user) \\ &\quad + \bigcup_{j=1}^{m} \bigcup_{i=1}^{n} Extension_i(C_e, group(user)_j) \end{aligned} \tag{5}$$

By adding the adjusted course (see formula 3) into this formula we obtain the course as a personalised, adjusted, and extended working course material as perceived by a user of the system:

$$C_{actual}(user) = C_{extended}(user) + C_{adjusted}(user) \tag{6}$$

3 Components

This section sketches the components of the SKILL system needed to provide the functionality described above.

Security Components For realizing the adjusted course contents and the personalised and shared course extensions a personal user identity is essential. For the rights managements of individuals and groups we are planing to use a role based access control system (RBAC) [11] which can be applied to web documents in a straightforward manner [8, 9]. The RBAC systems allows us to assign access rights to roles instead of users. This provides a convenient mechanism to manage the rights by separating the question of "which role needs access to which objects" and "which subject is in which role". By assigning a user to a role he/she is authorised to access all necessary objects. The basic roles are (1) the role of the students which allows them to use the course material and to extend the system with contents having an informational character (2) the role of the teachers which enables creating default courses and extensions and (3) the role of the system administrators. System users exist as individuals and group members. The user authentification will be realized by digital certificates [4] via SSL. The fact of authentification is important in allowing users to extend the system by annotations to documents or including new documents.

Document Management The comprehensive course materials are only manageable by a effective document management system. We decided to use XML [13] and CSS [12] as document structuring and layout definition languages since these standards proposed by the W3C are highly scalable, very flexible, and designed for the use on the Web. Furthermore version management will used as well as mechanisms to check link consistency. The current, HTML-based system uses CVS [3] for version management.

Tutoring Component The tutoring component provides a users' personal environment. It supports a personal configuration by storing a user profile. Furthermore it is responsible for logging the students' learning progress by storing visited concepts and exercise results. The tutoring component offers exercises in order to monitor and exercise the students learning progress.

4 Implementation

As mentioned above the SKILL project is an ongoing project which is not yet implemented. This section discusses various implementation issues that are entailed by the requirements.

All data belonging to the knowledge pool and all default navigation paths are stored in a area which is accessible for all registered users. Additionally every user and and every group has a home directory assigned which stores the private or group related extensions and annotations under a name with a reference to the source they are related to. The actual course navigation path is generated by a CGI-script which composes the path considering the default course, private and user group related extensions, and the learning progress. Handling the students learning progress can be done by using the browser-implemented concept of visited links in addition to the exercise results.

The implementation of the synthesis of the personalised documents from source documents can be realized by CGI-applications in the following way: For every request for a document a script is called which composes the personalised document from source document for all annotation sets in the relevant directories.

In order to store the position of an annotation in a document we will follow the approach proposed in [10], where the position of an annotation is stored together with the annotation itself. That means that the annotated document has no information about being annotated. The annotation needs the information about which document it is related to and at what position in the related document. This will be realized by a method named string position trees [10]. Each annotation is associated to the text part it is related to and to a position identifier string. Position identifier strings are defined as the smallest internal identifying string sufficient to locate a text position without ambiguousness.

In case the modification of the document breaks the position identification property, context information is kept as well. The context information is the textual context before and after the text position. The context information allows the repositioning of annotations in case the position identifier string mechanism fails in locating the annotations' position.

If both the smallest identifying string and the text context fail to locate the correct position, the annotation will be placed at the end of the document and marked as unassigned. Deviating from [10] the document merging procedure in the SKILL system is not implemented as a extend browser-feature but rather by an independent CGI-script usable by every standard browser.

The described functionality (adaptivity, annotations facility, etc.) could be implemented as a Java-applet or by CGI-scripts. In result of this trade-off we decided for the latter one. Realizing all necessary functions which provide adaptivity and annotation possibility in an Java-applet is maybe more comfortable to use, but means to disclaim the flexibility which is given by using CGI-scripts. An development basing on CGI-scripts which edit the files preserves the possibility to change the textual elements inside of an HTML/XML-file.

5 Conclusion and Related Work

The purpose of this paper is to describe the architecture of a teaching and learning system which allows user extensions, annotations to documents and a personalised configuration. Several of these functions are adressed by other software developments as well which have in parts similar purposes. The aim of a number of software developments (e.g. group-ware systems) is to provide a shared workspace that allows collaborative work. Using the WWW as a communication infrastructure is recommended

because (1) the WWW gained a broad acceptance, (2) WWW is easy to use and (3) the needed client software (browsers) and extensible server software exists for all important platforms.

One of the first applications using the web infrastructure to support collaboration work was the BSCW-system [2] developed at GMD in 1995. BSCW is based on the terminology of a shared workspace which is used as an information repository for various type of objects like documents, pictures, spreadsheets etc. BSCW users (in accordance with their access rights) upload, modify or remove objects from workspace, so it covers parts of the SKILL functions. The source code of the BSCW-server is freely available, can be integrated into SKILL and extended for further functions.

CoopWWW [1] is a project funded by the Telematics Application Programme which puts up on BSCW system. The goal of CoopWWW is to offer a set of tools supporting group cooperation using the BSCW system as kernel of this toolkit.

A further development in the area for collaboration work is OzWeb [5]. OzWeb is developed with the focus to support workflow enactment for distributed groups in a collaboration environment. OzWeb offers a tool service which is available on multiple platforms and allows common processing of a workflow.

The functionality offered by SKILL will combine collaboration aspects with individual adaptions. The focus lies on supporting learning and teaching processes. The initial SKILL development will be performed using the Cineast web browser [6] which is also a development at our department. Nevertheless we hope that most functionality will be available on all standard web browsers, once they have better XML and CSS support.

References

[1] W. Appelt, *CoopWWW – Interoperable Tools for Cooperation Support using the World-Wide Web*, http://orgwis.gmd.de/~appelt/coopwww.html, 03-04-98.

[2] W. Appelt, U. Busbach, *The BSCW System: A WWW-Based Application to Support Cooperation of Distributed Groups*, Proc. of the fifth Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Stanford 1996.

[3] Brian Berliner *CVS II: Parallelizing software development*, In Proceedings of 1990 Winter USENIX Conference, Washington, D.C., Winter 1990.

[4] CCITT Recommendation X.509, *The Directory–Authentication Framework*. Blue Book – Melbourne 1988, Fascicle VIII.8: Data Communication networks: Directory, International Telecommunications Union, Geneva, Switzerland 1989.

[5] G.E. Kaiser, S.E. Dossick, W. Jiang, J.J. Yang, and S.X. Ye, *WWW-based collaboration environments with distributed tool services*, World Wide Web, Vol. 1, No. 1, January 1998.

[6] E. Köppen, G. Neumann, *Cineast - An extensible Web Browser*, Proc. of WebNet 97, Toronto, Canada, November 1997.

[7] H. Maurer, *Hyper-G – now Hyperwave: The next Generation Web Solution*, Addison Wesley, Harlow 1996.

[8] G. Neumann, S. Nusser, *A Framework and Prototyping Environment for a W3 Security Architecture*, Proc. of Communications and Multimedia Security, Joint Working Conference IFIP TC-6 and TC-11, Athens, September 1997.

[9] S. Nusser, *Sicherheitskonzepte im WWW*, Springer, Berlin 1998.

[10] M. Röscheisen, C. Mogensen, T. Winograd: Beyond Browsing: Shared Comments, SOAPs, Trails, and On-line Communities, in Proceedings of the Third WWW Conference 1995, http://www-diglib.stanford.edu/diglib/pub/reports/brio_www95.html, 29-06-98.

[11] R. Sandhu, E. Coyne, H. Feinstein, and E. Youman, *Role-Based Access Control Models*, IEEE Computer, February 1996.

[12] W3C Proposed Recommendation 24-Mar-1998, *Cascading Style Sheets, Level 2*, http://www.w3.org/TR/PR-CSS2/, 22-06-98.

[13] W3C Recommendation 10-February-1998, *Extensible Markup Language (XML) 1.0*, http://www.w3.org/TR/1998/REC-xml-19980210, 22-06-98.