

Towards a Generic Adaptive System applicable for Web-based Learning Management Environments

Christian Gütl and Felix Mödritscher

IICM, Graz University of Technology

A-8010, Graz, Austria

{cguetl,fmoedrit}@iicm.edu

Abstract

One main objective of the AdeLE research project is the research and implementation of an enhanced, flexible and Web-based adaptive e-learning system by the application of fine-grained profiling and modeling information using multimodal sensor data in real time. Based on our knowledge about a wide range of existing learning management systems (LMS), we have decided to develop an approach to connect and control existing LMSs in order to extend them with adaptability or adaptivity. Backed on our experiences of the first fully functional implementation of the AdeLE system, we introduce our approach towards a generic adaptive system applicable for Web-based learning management environments in this paper.

1 Introduction

The power and flexibility of hypermedia for learning activities is as common knowledge as the known problems in this application area. Adaptive approaches are helpful solutions for problems in that context in a variety of application scenarios, which can supply personalized, tailored learning activities and methods to the learners.

In general, adaptive e-learning approaches are not new and date back to the early 1960s, see [Mödritscher et al., 2004a]. In recent years a lot of adaptive hypermedia systems (AHS) have been developed, virtually all of them fit into the adaptive hypermedia application model (AHAM) and its main aspects, as introduced in [De Bra et al., 1999]. Despite the long history in adaptive e-learning research and abundant implementations, from our point of view a number of open problems and requirements are not considered sufficiently by early as well as contemporary solutions. Therefore, we initiated the four-year research project AdeLE, see [García-Barrios et al., 2004] and [Guetl et al., 2004].

From a general point of view, the objective of AdeLE is to improve the learning process by well tailored learning activities in accordance to didactical constraints given by teachers or learning situations, and personal needs, such as preferred learning style and environmental constraints. Thus, the Web-based adaptive e-learning system has to support different adaptable features according to the learning path (sequencing), the content aggregation and the content representation.

Due to the fact that there exists a number of commercial or open-source learning management systems (LMS) and learning content management system (LCMS), the AdeLE project is intended to support existing systems. This allows content creators to choose their favored authoring tools and LMS administrators to run their favored

learning platform. Furthermore, reusability is also an important issue in that context.

In order to reuse learning material over system borders and application domains, at least the usage of small chunks of information, named learning assets, and its compilation against adaptation rules has to be supported. In addition, we propose an easy interchangeability and flexible reuse of domain knowledge and learning rules. Thus, the adaptive component has to deal with micro-adaptive instructional models and e-learning standards, see [Mödritscher et al., 2004a; 2004b].

The majority of available LMSs do not offer any adaptive features per se. Thus, the AdeLE system has to provide necessary adaptive features and offer a best practice to integrate existing LMSs with less implementation efforts. Furthermore, the adaptive component has to support different sets of functionality for each LMS. From the architectural point of view, the objective of AdeLE is to provide a platform-independent, flexible and open solution. Thus, the integration of additional modules and external services might easily be possible.

2 The AdeLE System at a Glance

To follow the line of main objectives and requirements of the project, the AdeLE architecture (see Fig. 1) is built up of strong separated client-side and server-side systems. A short overview is given focused on the server-side system components.

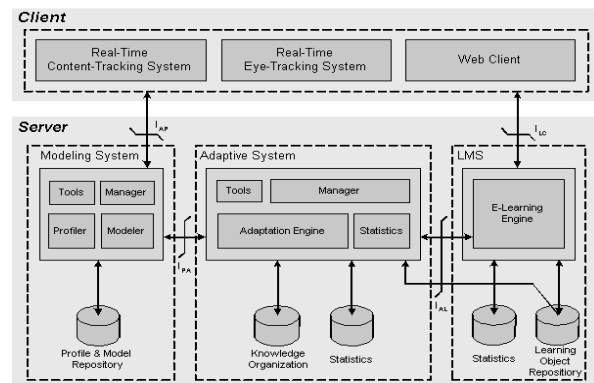


Figure 1: Architectural Overview of AdeLE

On the client-side, the Web Client (WC) – a simple Web browser – renders the delivered e-learning content and also provides control and navigation elements for interacting with the server-side located e-learning management system. In general, at the client-side numerous sensors could be applied to grasp information to be fed into the modeling system located on the server-side for gaining more fine-grained profiling information. At the current stage of the project only two input sensors at the client-side feed the modeling system. The Real-time Eye-Tracking System (ES) reads the eye movements and de-

livers characteristics of the user behavior. This information are linked with fine-grained content information of the learning assets rendered by the Web Client, which is processed by the Real-Time Content-Tracking System (CS). The first implementation supports the Microsoft Internet Explorers 6.x by applying a plug-in library and analyzing the eye-tracking parameters. The communication between the server and WC is handled by HTTP requests, synchronous RMI are applied for the remaining information exchange processes (see also section 4).

On the server-side, AdeLE's architecture is divided into three main systems: Modeling System (MS), Adaptive System (AS) and Adaptable System - in this case, a Learning Management System (LMS). The LMS compiles the learning content from the learning object repository, and provides the control and navigation interface. Adaptation of content, navigation and visualization is controlled by the AS by means of exploiting user information from the MS. The AS can also compile dynamically created content to be represented by the LMS. Because of the open architecture and an API provided by AS, existing LMS's can be integrated into the AdeLE environment; first reference implementation is built on the SCORM Runtime Environment of [ADL, 2005].

From the MS functional perspective, the main input tasks are to receive, log, manage and process user information from ET and CT, information about user interaction with the LMS and information about the adaptation processes by the AS. Output tasks of the MS are defined mainly by the generation of assumptions about the user and by the delivery of profiler insights.

The AS is located in the center on the server-side and it could be seen as the control center of the whole system. From the content creation point of view, the AS is in charge of adapting and personalizing learning activities by controlling the LMS or by composing and supplying dynamic content to the LMS. This functionality is carried out in accordance to the available LMS features, the domain knowledge about the courseware and learner's profile information. From the information flow point of view, the AS has to deal with information in different contexts. It has to receive and process information from the LMS about the users' interaction with the LMS, such as log in, clicking on a control button or select a learning activity. Such information as well as information about the adaptation processes by the AS also has to be sent to the MS system in order to build and update the fine-grained profile information. In contrary, the AS needs topically information from the MS about individual user sessions, such as the learner states.

3 Related Work

In literature and praxis there are a lot of reference models and realized systems for adaptive educational technology. This section gives a short review on chosen related work to point out differences compared in particular to AdeLE's adaptation model and adaptive component.

With respect to the Dexter model, the AdeLE system can be seen as a typical hypertext system realizing the three layers (storage layer, within-component layer, and runtime layer) described in [Halasz and Schwartz, 1994].

Compared to the Dexter model, some remarkable aspects can be found for our solution approach. The storage components are included in different nodes of the hypertext system and can be located even on different computer

systems. The components themselves are assembled on one or more nodes of the hypertext system. In contrary to other e-learning platforms, AdeLE provides an improved way to observe the user: (1) tracking the user interaction by means of the user's actions performed with the LMS, (2) by other sensors such as an eye-tracking device (see [García-Barrios et al., 2004]). Thus, information about the user is collected by two or more devices forwarded to and managed by a specialized node, the Modeling System.

Another more applied approach to AdeLE is the AHAM reference model described in [De Bra et al., 1999] and its related system named AHA!, see [De Bra and Ruiters, 2001]. In particular AHAM is of strong interest, because numerous system implementations fit well into this reference model. AHA! consists of a user modeler, the adaptive component and the presentation layer and is realized using web technology such as XHTML and Java Servlets. Links and content are adapted in terms of a domain and a user model connected with adaptation rules based on pre-defined concepts. In the AdeLE environment, all three types of models are available too – the user model is provided by the Modeling System, the course model by the LMS; adaptation rules are generated and managed by the Adaptive System. A domain model is also part of our system by means of a Concept Modeler. All in all we can identify the same elements as the AHA! system.

Yet, our architecture is based on a service-oriented approach, which allows us to be more flexible in terms of functionality and scalability: (1) The adaptive system can be combined with other LMS, too. (2) Specialized services for each task allow us to realize more sophisticated mechanisms for adaptation process; e.g. performing different types of adaptation or supporting other e-learning standards. (3) Our Modeling System provides different kind of models (learners, learner groups, etc.) on demand, and instead of just tracking user interactions with the hypertext system, AdeLE gains its fine-grained profile information by exploiting several sensor inputs. (4) Our distributed approach also allows integrating other external systems and tools into the adaptation process and we are able to adapt towards different devices.

The KnowledgeTree described in [Brusilovsky and Nijhawan, 2002] is a framework for adaptive e-learning based on distributed re-usable learning activities. This architectural model consists of a Student Model Server, a Portal and a different number of Activities Servers connecting these two components. During the learning process the portal uses the learning goal given by an activity server and the student model. The developed adaptive e-learning environment includes the student model server named CUMULATE as well as different portals, such as the KnowledgeTree or the KnowledgeSea. Activity servers are different educational services such as providing programming questions (QuizPACK) or adaptive demonstrations and exercises (WADEIn). The architecture, well documented in [Brusilovsky, 2004], is based on a distributed system supporting flexibility and scalability.

In fact, the KnowledgeTree concept is very similar to our approach. Mapping the architecture described in section 2 to this model, our LMS can be seen as the portal for the e-learning system. Thus, it is also our intention to exchange the portal, for instance by plugging our adaptive system to other e-learning platforms in order to provide adaptive educational techniques. The Student Modeling Server of the KnowledgeTree model can be seen as equi-

valent to our Modeling System, but our approach seems to be more general, which allows managing other models such as a context model, a domain model, and teacher model. The activity server of the AdeLE environment is meant to be the Adaptive System, but yet it is possible by our system to add other components for providing special services, e.g. other methods for adapting the learning process. A basic difference can be found in the communication model of the two technological frameworks. While the realized KnowledgeTree components use http-requests and xml-answers to exchange data, we are using synchronous RMI for the communication between the single components. Furthermore, we are using the SCORM specifications to model the course and to compile the learning content, while the KnowledgeTree realization is mainly based on CMI for describing learning content. Yet, both approaches, SCORM and CMI have their advantages and disadvantages as shown in [Conlan et al., 2002] and [Mödritscher et al., 2004b].

Being aware of a lot of other conceptual approaches and implementations of adaptive e-learning systems, we try to experience new concepts such as a hypertext system interacting with the user through different channels, developing a real distributed system to provide flexibility, extensibility and scalability, modeling the course on basis of an e-learning standard, supporting different LMSs, and using a common communication protocol.

4 Adaptive System Implementation Details

In general, the IICM's research work within the AdeLE project is focused on the server-side components, in particular in the development of the MS and the AS, as well as developing an approach to connect and control existing Web-based learning environments in order to extend them with adaptability. In order to follow the requirements stated in section 1, we have decided to develop both systems, AS and MS, on a Service-Based Architecture (SBA). In addition, we use two different techniques for both systems to gain further insights. MS is developed by means of what we call a micro-service approach (a lot of small and specialized services). AS is developed following our macro-service approach (a single service). The basic framework for the implementation of MS and AS is based on the Openwings framework specification, see [OPENWINGS, 2005]). Synchronous RMI is applied for communication between the systems.

4.1 LMS Integration into the AdeLE System

To follow the course towards a generic adaptive system applicable for a wide range of Web-based learning management environments, one emerging research task is to find an approach to the interaction between AS and LMS. Therefore, we propose a controlling and communication interface, which is provided by the AS and has to be applied by the LMS. The communication is based on synchronous requests from the LMS by the usage of a service request and a response object. The interface has to support the following features: (1) notify AS of any user action, (2) send information about the content and the course, such as course structure, domain knowledge, and didactical rules, to the AS, (3) control the LMS in order to adapt instructional sequencing, aggregation and representation, (4) return dynamic content to be displayed for the user, and (5) gain feedback from the learner via LMS using form-based content generated by the AS or PS.

In the context of the adaptation procedure we propose to support adaptation of the user interface (i.e. enabling or disabling control and navigation elements as well as providing or hiding a learning activity tree) and the learning content by content selection and content aggregation. Thus, we distinguish the adaptation process into three issues: (1) "sequencing" encompasses adaptation task to estimate the best learning activity in a particular context, (2) "aggregation" comprises adaptation tasks either to select the fittest content page from a set of pages or to compile a content page based on a set of content assets, and (3) "representation" summarizes adaptation tasks in order to present the learning environment and the content to particular needs for users and devices.

The following some subtasks can be applied for all issues stated above: (1) get proper model information, (2) adapt by course rules, (3) adapt by learner rules, (4) adapt by user feedback information, and (5) allow user to override the decision.

Because of reusability and the application of standards, our first implementation of the AdeLE system deals with SCORM-based content and the SCORM runtime environment (see also [ADL, 2005]) as an example for an external LMS. The first implementation of the interface between the adaptive system and the LMS encompasses the features encapsulated in three method: (1) "notifyAction": LMS notifies the Adaptive System about each user action, (2) "sendManifest": updates the cached course model in the Adaptive System, and (3) "sendFeedback": sends forms filled out and submitted to the Adaptive System. The current AdeLE system includes an error recovery strategy. If calling one of these methods fails – for instance because of missing network connectivity –, the LMS performs a silent catch and goes on "as usual".

Hence only three methods can be used by the LMS we did not intervene strongly in ADL's Sample Runtime Environment. We encapsulated this functionality into an own package and modified just a few source files of the original LMS. Actually, we can report about five modified source-files – all of them are JSPs, four of them only have been slightly extended, e.g. by the call to notify the Adaptive System about a certain user action. Yet, there is one JSP page called sequencing engine, which mainly covers the learning process and, therefore, is affected strongly by AdeLE-specific modifications.

We believe that the proposed approach for integrating learning management environments might be applicable for a wide range of different systems and points to the right direction towards a generic solution.

4.2 Architectural and Software Design

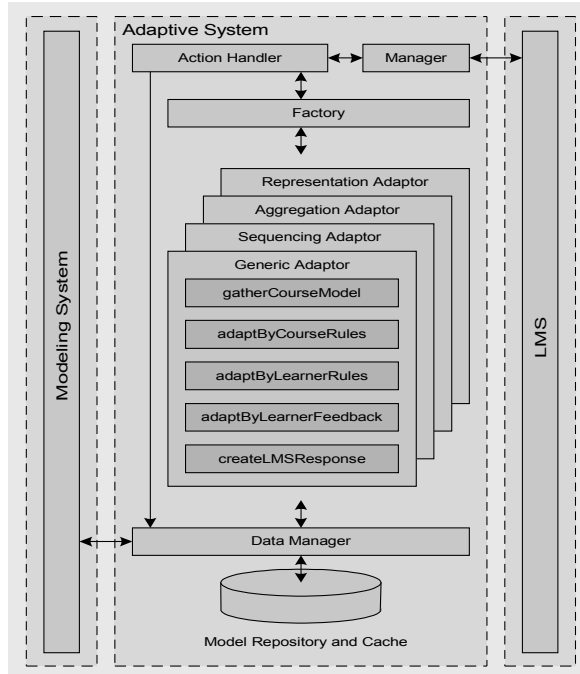
The detailed architecture of the Adaptive System is shown in figure 2, whereby the first implementation is also designed for SCORM-based learning management systems.

The Manager and the Action Handler process the incoming requests performed by the LMS. Based on the type of method call and the content of the communication object, the request is passed to an Adaptor generated and managed by the Factory and particular information are passed to and retrieved by the Data Mangers, such as information about the user interaction with the LMS, adaptation decisions by the AS, and the learning behavior acquired by the eye-tracking device. Due to the fact that the Adaptive System has to manage the didactical rules

and communicate with the Modeling System to store and retrieve learner-dependent adaptation rules, we designed and implemented some kind of data-layer for these features. The Data Manager offers a set of important database-operations on basis of an own abstract data-handler as well as a collection of methods accessing the Modeler System – both components are implemented as singleton to guarantee data consistency and a better performance.

Figure 2: Architecture of the Adaptive System

The adaptation of certain aspects in the LMS – such as calculating the next instruction, aggregating or presenting a chosen instruction, and so forth – is performed by so-called Adaptors. A Generic Adaptor consists of 5 necessary methods always executed in the same order: First of all, the adaptor tries to retrieve the course model, which is the DOM representation of the SCORM-based course’s manifest file. For further use the course model is cached and is also sent to the MS for archiving purpose. Secondly, the Adaptor attempts to adapt the depicted aspect of the LMS with respect to didactical rules – this operation may result in a certain decision, which is saved in the Model Repository to be reused in the context for this particular course. The third step describes the personalization process itself, where the adaptor tries to adapt the chosen aspect of the LMS for the learner on basis of the user model given by the Modeling System. If this does not work, the Adaptor generates a feedback form to ask the user to complete the adaptation step. Hence, the system’s or the learner’s decision is stored as an own rule within the Modeling System (in the user model) and can be reused to reproduce the adaptation process for this certain user, to provide recommendations to other learners or to analyse the course content or the learning process. Finally, the Adaptor always generates a response object to control the LMS. This abstract Adaptor is implemented for sequencing, aggregation and representation using the Factory pattern. The three adaptors follow the proposed adaptation process discussed in the section above.



Although it is easy to implement new Adaptors, our solution is not perfect yet. The realized Adaptors are highly dependent on the LMS which has to be a Servlet-based system able to import SCORM packaged courses. Furthermore, it would be more powerful and advantageous to use the Abstract Factory pattern to allow the exchange of the whole product family (SequencingAdaptor, AggregationAdaptor, RepresentationAdaptor), for instance to supply another LMS or standard. Fortunately, this suggestion does not affect the provided interfaces and services to other systems and, therefore, can be realized later on. Yet, we are about to find a more general way to generate, to manage and to exchange families of Adaptors, not only for the e-learning situation, but also to extend any kind of system with an adaptive component.

To clarify the interaction between the LMS, the AS and the MS, an example is discussed in the remainder of this section. Figure 3 visualizes a sequence diagram for the use case “show instruction” and, therefore, describes how the learner and the three involved systems communicate with each other. The communication between the systems has been simplified by drawing and labelling the answers of the synchronous method calls.

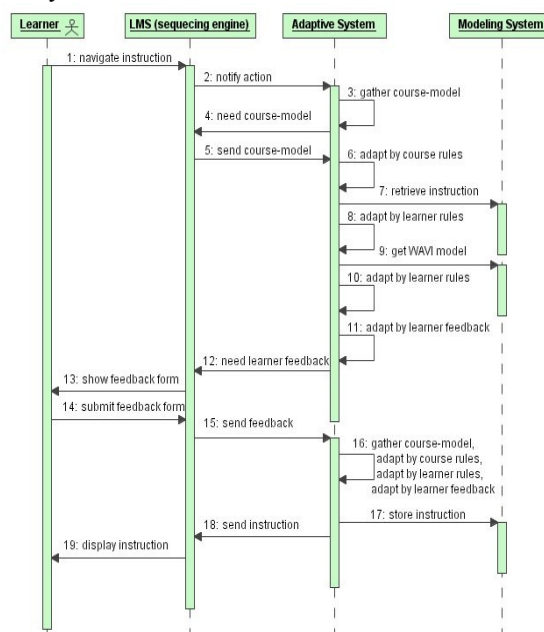


Figure 3: Sequence diagram for use case “show instruction”

If a user performs a new action to go on within the learning process, for instance by pressing the next button or choosing an instruction in the tree-view, the LMS informs the AS about this interaction and waits for an answer, the response object. If the Adaptive System is not running or some kind of network error happens, the LMS continues as if there is no other system to rely on. Yet, in the AdeLE framework the Adaptive System performs the adaptation process as explained before and returns one of the following answers:

(a) The Adaptive System might return the best instruction to go on, if the next button was pressed and an adaptation of the instructional sequencing was performed. In fact, the list of the instructions is reorganized with every user action in the background, although it is not visible at any time (not treated in figure 3).

(b) The Adaptive System can also send back the best fitting resource in the SCORM package to display, if it adapts the learning process by means of content aggregation or presentation as described at the end of the sequence diagram in figure 3 (request 18 and 19).

(c) Furthermore, the Adaptive System or the Modeling System might miss some information from the LMS. In that case, the Adaptive System can return control commands to the LMS. Request 4 and 5 in figure 3 demonstrate how the course model is requested by the Adaptive System.

(d) The Adaptive System or the Modeling System may be in need of learner feedback. Therefore, these systems can generate forms to be filled out and submitted by the learner as shown in figure 3 (request 12 to 15).

(e) Finally, the Adaptive System can return self-generated HTML fragments to be visualized by the LMS.

4.3 Models and Rules

In order to operate the adaptation processes highlighted in 4.1, the AdeLE system relies on a set of models and on a set of rules. In a general solution's point of view, our first theoretical research approach has incorporated the following models: (1) domain knowledge for modeling the subject, concepts and its relations, (2) course model for describing the course structure and links to content assets and its alternatives in order to different media types and knowledge level, (3) didactic model for describing a particular didactic and its objectives, and finally (4) the user model for representing users' attributes and their behaviours. Based on these models, the set of rules defines how to deal with the model information, i.e. the rules define how to interpret particular information and use it for the adaptation process.

A more practical approach of the adaptation process for the AdeLE system is described in [Mödritscher et al., 2004a] and establishes a basis for our first implementation of the system. Starting point of the adaptation process is the course model, which describes the sequencing (possible paths through the learning activities) as well as possible sets of content. At this step the system deals with a set of undecided choices. A didactic adaptation against a particular didactic scenario has to be performed in order to provide a starting point for the personalization. The adaptation on basis of the learner model takes place. If a final decision is not possible by means of the user model, i.e. no unique solution could be found, the learner can be asked by the MS or the AS to complete the adaptation process. The first implementation for the SCORM

standard based on the basic adaptation concept stated so far is described in the remainder of this section.

Thus, the Adaptive System is in need of a course model which we derived by the SCORM standard (see [Mödritscher et al., 2004b]). This course model is requested by the LMS and stored in the Model Cache (see figure 2). Let us consider that a student has finished a particular content page and wants to get the next learning instruction. Having a SCORM manifest file with some unresolved decisions, such as several instructions to go on or different resources for one instruction, the AS performs the first adaptation step, the so-called didactical adaptation. Resulting from this first step, we receive some kind of didactical rule for this course. A didactical adaptation rule can be seen as a mapping of the course-id to the concrete adaptation result, for instance an ordered list of instructions to go on or the appropriate resources aggregated for an instruction. As this rule is only dependent on the course, it is saved within the Model Repository to reuse it for all other students entering the same course. This adaptation process deals with sequencing and aggregating instructions and is performed by the sequencing adaptor and the aggregation adaptor. In our current implementation the corresponding rules are implemented as some sort of methods in the sequencing adaptor. Adapting the instruction's presentation is planned and taken in consideration, but not realized in the first prototype.

The above mentioned didactical rule might not be the final decision – some adaptation options could still remain. Therefore, in the second adaptation step the Adaptive System tries to adapt one chosen aspect with respect to the learner model given by the Modeling System. Our central user model at this time is the WAVI model, see [Riding, 1991], where we use the wholist-analyst factor to adapt the LMS's tree-view and the verbalizer-imager factor to provide more text passages or visual assets. Failing this personalization process, the Adaptive System continues by simply asking the user for the final decision. The resulting rule, e.g. the personalized instruction, is sent to the Modeling System and displayed in the LMS. Furthermore, the Modeling System could update the user model on the basis of the Adaptive System's decision or the learner's feedback. This step of the adaptation process is also performed by the sequencing adaptor and the aggregation adaptor, and again the set of rules are implemented as some sort of methods in these adaptors.

5 First Experiences

First of all, we can recommend a distributed architecture as well as the Openwings framework for AdeLE's research and development approach. On the one side, our approach allows utilizing of existing LMSs, such as ADL's exemplary runtime environment. On the other side, it seems to be promising to separate the Adaptive System and the Modeling System in terms of flexibility and modularity. Providing generalized services by these two systems would simplify implementing an adaptive or modeling component for any other application.

With respect to the communication model, we decided to use synchronous communication between the systems, because it is fast to implement and easier to understand. But we face some disadvantages: On the one side, this solution harms the aspect of modularity, for instance by

the AS's response object which requires the LMS to know about how to react to the response. On the other side, the adaptation process itself is triggered by the LMS and the sensor devices. The AS and MS only perform internal data processing, if they got a request by an outer system. Furthermore, it is not possible that the MS uses a service of the AS or the AS of the LMS – such a circle is not allowed for synchronous communication. A solution to this problem would be applying asynchronous.

Regarding the architectural design, we experienced both, the macro- and the micro-service approach. On the one side, the micro-service approach causes a communicational overhead and is very complex due to a larger number of services communicating with each other. We are aware of the danger of a partial or full system breakdown, e.g. if one service fails or is not available. On the other side, the macro-service approach allows the extended usage of design patterns and, therefore, a compact design for a system consisting of a single service. Nevertheless, this approach is very inflexible in terms of reusability of functionality. A good solution appears to be in the middle of these two approaches. To reduce communication efforts by numerous services, we are also planning to realize smart caching mechanisms for user profile information similar to the cache for the course model.

Concerning the design and implementation of the Adaptive System, we still face some unsolved problems: (1) A more general model in order to support different LMSs and standards without realizing the rules by program code is needed. (2) We still did not decide, if the course model and the didactical rules should be managed and provided by the Modeling System. (3) Concerning the architectural design of the AS, it would be better to implement an abstract factory to exchange a whole adaptor family at once. (4) The interface of the abstract adaptor has to be reconsidered to adapt the adaptors for any LMS or standard. (5) The interface between systems to be adapted and the adaptive system – and the response mechanism for synchronous communication – has to be designed more general to provide adaptation of general aspects of any system.

6 Conclusions

Summarizing the approach of our first prototype, the adaptation process is highly dependent on the SCORM specifications. In fact, we are aware of this situation, but considering other standards and specifications, we always find the same basics somehow: AICC (see [AICC, 2005]) and IMS (see [IMS, 2005]) also have specifications to sequence and to aggregate the instructions. Therefore, our theoretical model of adaptation is that general to be able to adapt to courses based on these two specifications, too. With respect to our inquiry on adaptive e-learning, we can reduce the important and not content-specific aspects of the learning process to sequencing, aggregating and presenting the instructions. All other important factors of adaptive e-learning are somehow strongly related with the learning content itself.

From our point of view it is worth to intensify our research work in order to enhance the features and flexibility of the AS and MS. Thus, we plan to release our development as open source in order to get further input and developer power. As a further next step we will work on generalized MS and AS to be able to apply the systems also in further application domains.

1 Acknowledgments

The AdeLE project is partially funded by the Austrian ministries BMVIT and BMBWK, through the FHplus impulse program. The support of the following institutions and its individuals is gratefully acknowledged: Department of Information Design, Graz University of Applied Sciences (FH JOANNEUM); Institute for Information Systems and Computer Media (IICM), Graz University of Technology.

2 References

- [ADL, 2005] ADL. Advanced Distributed Learning, 2005. <http://www.adlnet.org> (2005-07-15)
- [AICC, 2005] AICC. Guidelines and Recommendations, Version 3.0, 2005. <http://www.aicc.org/pages/down-docs-index.htm#AGR> (2005-07-15)
- [Brusilovsky, 2004] Peter Brusilovsky. KnowledgeTree: A distributed architecture for adaptive e-learning. In *Proceedings of the International World Wide Web Conference (WWW 2004)*, pages 104-113, New York, USA, 2004.
- [Brusilovsky and Nijhawan, 2002] Peter Brusilovsky and Hemanta Nijhawan. A framework for adaptive e-learning based on distributed re-usable learning activities. In *Proceedings of World Conference on E-Learning (E-Learn 2002)*, pages 154-161, Montreal, Canada, 2002.
- [Conlan et al., 2002] Owen Conlan, Declan Dagger and Vincent Wade. Towards a standards-based approach to e-Learning personalization using reusable learning objects. In *Proceedings of World Conference on E-Learning (E-Learn 2002)*, pages 210-217, Montreal, Canada, 2002.
- [De Bra and Ruiters, 2001] Paul De Bra and Jan-Peter Ruiters. AHA! Adaptive Hypermedia for All. In *Proceedings of World Conference of the WWW and Internet (WebNet 2001)*, pages 262-268, Orlando, USA, 2001.
- [De Bra et al., 1999] Paul De Bra, Geert-Jan Houben, and Hongjing Wu. AHAM: A Dexter-based Reference Model for Adaptive Hypermedia. In *Proceedings of 10th ACM Conference on Hypertext and hypermedia (HT 1999)*, pages 147-156, Darmstadt, Germany, 1999.
- [García-Barrios et al., 2004] Victor Manuel García-Barrios, Christian Guetl, Alexandra Preis, Keith Andrews, Maja Pivec, Felix Mödritscher, and Christian Trummer. AdELE. A Framework for Adaptive E-Learning through Eye Tracking. In *Proceedings of IKNOW 2004*, pages 609-616, Graz, Austria, 2004.
- [Guetl et al., 2004] Christian Guetl, Victor Manuel García-Barrios, and Felix Mödritscher. Adaptation in E-Learning Environments through the Service-Based Framework and its Application for AdeLE. In *Proc of E-Learn 2004*, pages 1891-1898, Washington, USA, 2004.
- [Halasz and Schwartz, 1994] Frank Halasz and Mayer Schwartz. The Dexter Hypertext Reference Model. *Communications of the ACM*, 37(2): 30-39, February, 1994.

[IMS, 2005] IMS Global Learning Consortium. Specifications, 2005. <http://www.imsglobal.org/specifications.html> (2005-07-15)

[Mödritscher et al., 2004a] Felix Mödritscher, Victor Manuel García-Barrios, and Christian Guetl. The Past, the Present and the Future of adaptive E-Learning. An Approach within the Scope of the Research Project AdeLE. In *Proceedings of ICL 2004*, Villach, Austria, 2004.

[Mödritscher et al., 2004b] Felix Mödritscher, Victor Manuel Garcia Barrios, and Christian Gütl. Enhancement of SCORM to support adaptive E-Learning within the Scope of the Research Project AdeLE. In *Proceedings of World Conference on E-Learning (E-Learn 2004)*, pages 2499-2505, Washington, USA, 2004.

[OPENWINGS, 2005] OPENWINGS. Official Website, General Dynamics Decision Systems, 2005. <http://www.openwings.org> (2005-07-15)

[Riding, 1991] Richard Riding. *Cognitive Style Analysis Users' Manual*, Birmingham Learning and Training Technology, 1991.