Learner Profile Management for Collaborating Adaptive eLearning Applications

Mohammad Alrifai
L3S Research Center
University of Hanover
Expo Plaza1
D-30539 Hannover
Germany
alrifai@l3s.de

Peter Dolog
L3S Research Center
University of Hanover
Expo Plaza1
D-30539 Hannover
Germany
dolog@l3s.de

Wolfgang Nejdl
L3S Research Center
University of Hanover
Expo Plaza1
D-30539 Hannover
Germany
nejdl@l3s.de

ABSTRACT

Adaptive Learning Systems would perform better if they would be able to exchange as many relevant fragments of information about the learner as possible. The use of Web Services standards is recently gaining the attention of many researches as a promising solution for the problem of interfacing adaptive hypermedia systems. Existing Web Service standards, however, only provide very basic features and leave out many important issues like transactional management. In this paper we propose a mechanism for enabling consistency maintenance of Learner Profiles shared between collaborating adaptive learning systems.

Categories and Subject Descriptors

H.5 **[Information Interfaces and Presentation]:** Group and Organization Interfaces – *Collaborative computing, Web-based interaction.* H3.4 **[Systems and Software]**: User profiles and alert services. H3.5 **[Online Information Services]**: Web-based Services.

General Terms

Design, Management, Standardization.

Keywords

Adaptive Systems, eLearning Systems, Learner Profile, User Models, Web Services, Consistency Management.

1. INTRODUCTION

In the last years several learning systems have been developed trying to find a solution to the problem of personalization of the learning process. They aim to "learner centered education" by providing an adaptive learning system which is able to adapt the course to the learner's specific characteristics, knowledge, objectives and learning goals as well as is capable of adjusting the appropriate parameters according to the context in which the learning session is taking place.

Learners are assessed by several systems during their professional learning. Those systems can maintain fragments of information about a learner derived from her learning performance and/or assessment in that particular system. Several approaches have been proposed to collect information about users such as preferences, following clicking behavior to collect likes and dislikes, and questionnaires asking for specific information to assess learner features (e.g. tests, learner assessment dialogs, and preference forms). In addition, several tools have been designed to improve learner models by open active learner modeling. The variety of use cases are supported by such tools like maintaining and comparing the student's own and the system's believes about his knowledge [8], multiple choice questionnaires [7], collaborative peer assessment in discussions [6], and dialogues with interactive topic maps [9].

Adaptive Learning Systems would perform better if they would be able to exchange as many relevant fragments of information about the learner as possible. Therefore, there is a necessity to combine different user models into interoperable Learner Profiles. A recommended solution for the interoperability and interfacing problem is to work with standardized technologies and formats as far as possible, especially Semantic Web standards (e.g. RDF, OWL) for modeling exchangeable Learner Profiles and Web Services standards (e.g. SOAP, WSDL, UDDI) for interfacing collaborating adaptive systems. In [1], [2] and [3] Dolog et al. suggest how modeling exchangeable Learner Profiles can be accomplished through the use of Learner Ontologies and RDF [4] [5]. In [3] Dolog et al. give an example of how three different Learner Information data models are used to describe an Ontology-based, exchangeable Learner Profile, focusing on learner performance.

The use of Web Services is gaining the attention of several researchers as a promising solution for the problem of interfacing loosely coupled collaborating adaptive systems. Existing Web Service standards, however, only provide very basic features and leave out many important issues like transactional management.

In this paper we attempt to address the following question:

APS'06, August 23, 2006, Odense, Denmark.
Copyright 2006 ACM 1-59593-453-7/06/0008...$5.00.

- *how consistency of Learner Profiles is maintained between concurrently running web service-based collaborating adaptive learning applications?*

We extend a framework and API for management of distributed learner profiles we have developed and described in [2,3] with a protocol, which will ensure a consistency of the profile used in the collaborating adaptive applications.

In Section 2 we describe a motivating scenario and point out the shortcomings of the use of Web Services in the context of exchanging interoperable Learner Profiles. In Section 3 we describe our Learner API and in Section 4 we propose an extension to this API to cope up with the problems discussed in Section 2. Section 5 introduces more details about the proposed Learner Profile Consistency Management Protocol. Section 6 concludes this work and gives an outlook to future work.

2. MOTIVATING SCENARIO

Figure 1 depicts a scenario of how to access and exchange learner profile fragments based on the model described in [3]. The fragments can be accessed by web services, which export the learner model and acts as a Learner Profile (LP) server. The web services provide functions for getting, deleting and updating a model of the fragment. Application/LP server interfacing web services handle all requests from the LP clients.

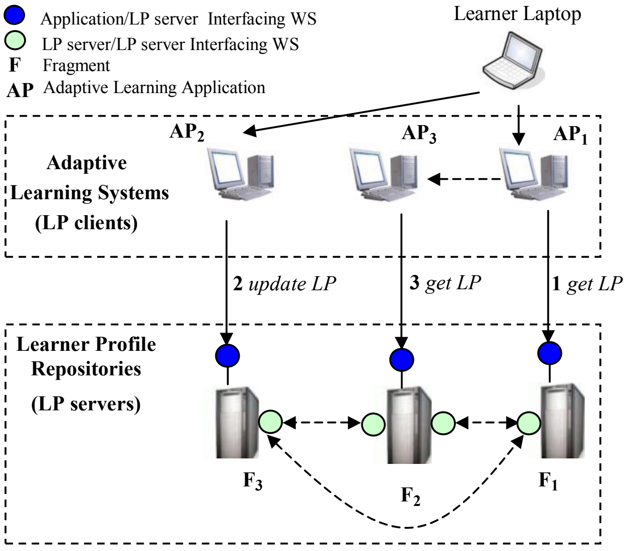
● Application/LP server Interfacing WS
○ LP server/LP server Interfacing WS
F Fragment
AP Adaptive Learning Application
Learner Laptop
AP_2
AP_3
AP_1
Adaptive Learning Systems (LP clients)
2 *update LP*
3 *get LP*
1 *get LP*
Learner Profile Repositories
(LP servers)
F_3
F_2
F_1

Figure 1. The use of the Web Services for Exchanging Learner Profiles

They perform as *ontology mediators*, i.e. for resolving mismatches between different used terminologies in the Learner Information level. Each client is uniquely identified at the server and can be used by a browsing or assessment system. Furthermore, a client can be used by other learning systems which want to make use of the learner profiles or which want to contribute to them. LP servers can exchange their local learner information with each other to enrich their Learner models. Each server maintains a list of LP Servers and the fragments maintained by each one. When a LP server receives a request from a LP client (adaptive application), it retrieves the learner information fragments from the hosting LP servers and returns an enriched learner profile to the requester. LP servers use web services to communicate with each other (LP server/LP server interfacing web services).

According to this scenario, several clients can access one model, which is distributed on several servers by invoking functions of the servers' web services. Figure 1 illustrates an example scenario where a learner is concurrently participating in several adaptive learning applications. The numbers on the arrows represent the sequence of the access actions made to the learner profile. In Step 1, the first application AP_1 requests the learner profile by invoking the *get* function of the web service on the LP server holding the fragment F_1 of the learner profile. The invoked web service in turn retrieves the learner information from the different fragments, builds a learner profile and returns it to the requesting client. In Step 2, the second application AP_2 contributes to the learner profile and updates F_3 by invoking the update function on the corresponding LP server web service. In Step 3, AP_3, used by AP_1, retrieves the learner profile by invoking the web service on F_2. At this point, it turns out that AP_1 and AP_2 (both applications involved in one learning process) have different views to the learner profile.

Such WS-based systems should guarantee consistent data manipulation. In contrast to traditional transactions, WS transactions live long, e.g. a few hours or days. Thus, if the strict locking mechanisms are used, an organization may not be able to access its resources even for a few days, until other organizations complete their works and release their resources. This also holds for Adaptive Learning Systems sharing Learner Profiles resources where locking learner profile fragments until the end of a learning process is undesirable.

Therefore, Web Services technologies should be extended to equip with transaction-processing functionalities. There are three proposals for protocols to extend the Web Services with transaction processing capabilities, i.e., Web Services Transactions specifications [10], Business Transaction Protocol (BTP) [11], and WS-CAF [12]. For efficient processing, these WS-based transaction protocols relax the isolation property rather than using strictly exclusive locking mechanisms such as the two-phase commit and the strict two-phase locking [13]. However, these protocols do not handle the inconsistency problem of isolation-relaxing transactions.

3. LEARNER PROFILE API

The learner profile API is derived from the ontology for learner profiles. It provides access classes to access and manipulate learner profile fragments about particular user feature represented by a concept from the ontology.

Figure 2 depicts an excerpt of a learner profile ontology configured from fragments based on three specifications[1]. The conceptual model describes a situation where a learning performance of a student is exchanged as his achieved competency records. The competencies have been evaluated by learner assessment (e.g. tests) and were derived from learning

[1] http://www.l3s.de/~dolog/learnerrdfbindings/

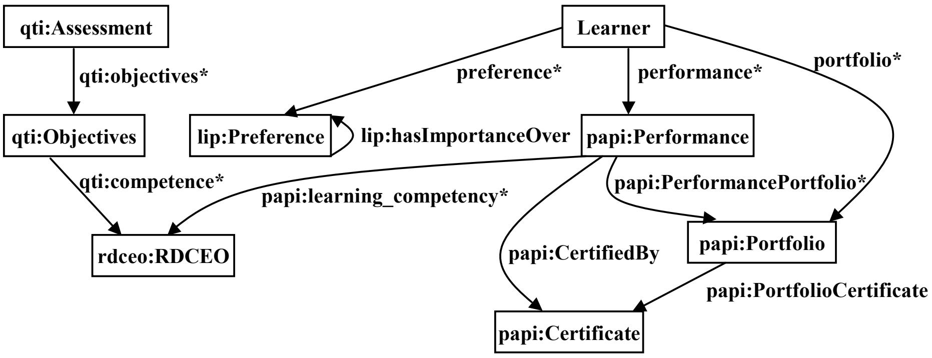

Figure 2: An excerpt of a conceptual model for learner profile based on standards

objectives of tests. Furthermore, all other educational activities, further materials, and projects created within the activities are reported within the portfolio of the performance. Additional information, which is reported under preferences, comprises language, device, resource and learning style preferences. The standards and open specifications guarantee wider acceptance between eLearning systems and as such can be seen as good candidates for the learner exchange models. The API contains classes similar to the ones depicted in the model.

4. LEARNER PROFILE CONSISTENCY MANAGEMENT

In our proposal we assume the following: 1) each Learner profile is uniquely identified, 2) learner information can be distributed among several fragments, 3) each server keeps a mapping between a Learner Profile ID and servers maintaining the relevant fragments. These assumptions are consistent with the API we have employed for our experiments.

To ensure a consistent Learner Profile view among concurrently running collaborating adaptive systems, each LP server should be able to notify the *active* adaptive systems about any updates made to the Learner Profile being used by them. Therefore, adaptive applications should inform the LP servers about the start and the end of the learning process they are involved in to enable the LP servers to keep track of all *active* applications and the LPs used by them. This can be achieved for example by subscribing/unsubscribing from the notification service provided by the LP servers.

In the example scenario given in Figure 1, this means that the server holding F_3 should notify the server holding F_1 about the update action made to F_3 by AP_2 and the server holding F_1 in turn should notify AP_1 about this update (if AP_1 has already registered for the notification service). Upon the receipt of this notification, AP_1 (optionally) uses the new version of the Learner Profile fragment and updates its own Learner Model accordingly. Thus, after Step 3, both AP_1 and AP_3 will have the same view over the Learner Profile of the learner served by them.

5. CONSISTENCY MANAGEMENT PROTOCOL

To realize the proposed Learner Profile Consistency Management Protocol, we extend the framework described in Section 2 (Figure 1) to support some additional operations and messages. More precisely, the LP server/LP server interfacing services should support the following operations:

SendLP operation: this operation is used to request a learner profile fragment from the LP server holding this fragment. It is invoked by sending a SOAP message: *SendLP* message (Table 1) including the LP_ID of the required learner profile and indicating whether the requester wants to register for notification service about any updates to this learner profile fragment. Upon the receipt of this message, the server sends its local learner information to the requester in a SOAP message.

ReceiveLP operation: this operation is invoked (on the requester side) by sending a SOAP message: *ReceiveLP* message (Table 2) including the requested learner fragment. This operation is invoked either as a response to a previous explicit request (through a *SendLP* message) or upon an update action to the relevant learner information fragment. Only those who are registered for notification service about this learner profile will receive the updated version.

StopNotification operation: this operation is used to unsubscribe from the notification service for a particular learner profile and is invoked by receiving a *StopNotification* message (Table 3).

Figure 3 shows the WSDL description of the operations mentioned above.

```
<wsdl:portType name="ExchangeLPPortType">
        <wsdl:operation name="SendLP">
                <wsdl:input message="lp:LPRequest"/>
        </wsdl:operation>
        <wsdl:operation name="ReceiveLP">
                <wsdl:input message="lp:LPResponce"/>
        </wsdl:operation>
       <wsdl:operation name="StopNotification">
                <wsdl:input message="lp:StopNotification"/>
       </wsdl:operation>
 </wsdl:portType>
```

Figure 3: LP server/LP server interfacing Web Service

Table 1: Elements in a *SendLP* Message

Element	Description
LP_ID	The unique ID of the requested learner profile
NOTIFY	TRUE, if notification about updates is required, otherwise FALSE

Table 2: Elements of a *ReceiveLP* Message

Element	Description
LP_ID	The unique ID of the sent learner profile
LP	The content of the learner information

Table 3: Elements of the *StopNotification* Message

Element	Description
LP_ID	The unique ID of the learner profile

6. DISCUSSION

In the proposed protocol we use a separated function (*ReceiveLP*) for delivering the learner profile instead of sending it as a response (output message) to a *SendLP* request. This is to enable sending updated versions of the learner information fragments to the appropriate clients in an asynchronous (event-based) manner, i.e. whenever an update is made to the relevant learner profile fragment.

To implement this protocol, it is necessary for each LP server to maintain a mapping between LP servers, which are registered for notification service and the LP_IDs of the learner profiles they are registered for. Upon the receipt of any *SendLP* or *StopNotification* message, this mapping should be updated accordingly.

It is worthy to mention that for some adaptive learning systems, which update the learner information very frequent during the learning session, the number of the exchanged messages (particularly the *ReceiveLP* message) will increase significantly. Thus the load produced by these messages might be an issue. However, in the proposed protocol, adaptive applications *optionally* subscribe for a notification service and once they did, they still have the opportunity to unsubscribe from this service whenever they want.

7. CONCLUSION

In this paper, while encouraging the use of web services for interfacing loosely coupled collaborating adaptive learning systems, we also point out to the shortcomings of the use of Web Services with respect to learner profile consistency management. To address this problem, we extend a framework for management of distributed learner profiles we have developed with a protocol, which will ensure a consistency of the profile used in the collaborating adaptive applications. We are now working on integrating the extended functions to our current Learner API.

8. REFERENCES

[1] Dolog, P., & Nejdl, W. (2003, June 2003). Challenges and Benefits of the Semantic Web for User Modelling. Paper presented at the AH2003 Workshop, User Modelling Conference.

[2] Dolog, P. (2004). Identifying Relevant Fragments of Learner Profile on the Semantic Web. Paper presented at the Workshop on Applications of Semantic Web Technologies for E-learning - SW-EL'04: Semantic Web for E-Learning at International semantic Web Conference: ISWC 2004, Hiroshima, Japan.

[3] Dolog, P., & Schäfer, M. (2005). A Framework for Browsing, Manipulating and Maintaining Interoperable Learner Profiles. In Proc. of UM2005 - 10th International Conference on User Modeling, July, 2005, Edinburgh, UK. Springer Verlag.

[4] Hayes, P. (2002). RDF Model Theory. Retrieved February 14, 2003,from http://www.w3c.org/TR/rdf-mt/

[5] Nilsson, M. (2001). The Semantic Web: How RDF will change learning technology standards. Feature article, Centre for Educational Technology Interoperability Standards (CETIS).

[6] Susan Bull, Paul Brna, Sonia Critchley, Koula Davie, and Corina Holzherr. The missing peer, artificial peers and the enhancement of human-human collaborative student modelling. In S.P. Lajoie and M. Vivet, editors, Proc. of International Conference on Artificial Intelligence in Education, pages 269-276. IOS Press, 1999.

[7] Susan Bull and Theson Nghiem. Helping learners to understand themselves with a learner model open to students, peers and instructors. In Paul Brna and Vania Dimitrova, editors, Proc. of Workshop on Individual and Group Modelling Methods that Help Learners Understand Themselves, International Conference on Intelligent Tutoring Systems, pages 5-13, 2002.

[8] Susan Bull and Helen Pain. Did I say what I think I said, and do you agree with me? In J. Greer, editor, Proceedings World Conference on Artificial Intelligence in Education, AACE, pages 501-508, 2002.

[9] Vania Dimitrova. Style-olm interactive open learner modelling. International Journal of Artificial Intelligence in Education, 13, 2002.

[10] IBM, Microsoft, and BEA. Web Services Transactions Specifications. http://www-106.ibm.com/developerworks/webservices/library/ws-transpec/

[11] OASIS. Business Transaction Protocol. http://www.oasis-open.org/committees/documents.php?wg_abbrev=business-transaction

[12] Arjuna Technologies Ltd., Fujitsu Software, IONA Technologies PLC, Oracle Corp, and Sun Microsystems. Web Services Composite Application Framework (WS-CAF). http://developers.sun.com/techtopics/webservices/wscaf

[13] J. Gray and A. Reuter. Transaction Processing: Concepts and Techniques. Morgan Kaufmann Publishers.