Making today's Learning Management Systems adaptive

Lucia Oneto[1], Fabian Abel[2], Eelco Herder[2], David Smits[3]

[1] Giunti Labs, Abbazia dell'Annunziata, Via Portobello, 16039 Sestri Levante, Italy
l.oneto@giuntilabs.com
[2] L3S Research Center, Appelstr. 9a, 30167 Hannover, Germany
{abel,herder}@L3S.de
[3] Faculty of Mathematic and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
d.smits@tue.nl

Abstract. The research and development of Learning Management Systems (LMS) and Adaptive Learning Environments (ALE) remained disconnected for a long time. The most popular LMSs like Moodle, Sakai or Blackboard still do not support personalization as found in existing adaptive educational hypermedia systems and applications. This document reports the experience and efforts to bridge the gap between LMSs and ALEs by bringing adaptation and personalization into LMSs, including adaptation to individual goals, knowledge, learning styles and context of use as well as adaptation to group characteristics and goals and tasks of groups doing collaborative work.

Keywords: elearning, learning management systems, adaptation, integration, personalization, user modeling

1 Introduction

A Learning Management System (LMS) is software for delivering, tracking and managing training/education. LMSs range from systems for managing training/educational records to software for distributing courses over the Internet and offering features for online collaboration. In many instances, corporate training departments purchase LMSs to automate record-keeping as well as the registration of employees for classroom and online courses. Student self-service (e.g., self-registration on instructor-led training), training workflow (e.g., user notification, manager approval, wait-list management), the provision of on-line learning (e.g., Computer-Based Training, read & understand), on-line assessment, management of Continuous Professional Education (CPE), collaborative learning (e.g., application sharing, discussion threads), and training resource management (e.g., instructors, facilities, equipment), are dimensions to Learning Management Systems.

Most LMSs are web-based to facilitate access to learning content and administration. LMSs are used by regulated industries for compliance training and by

educational institutions for enhance and support classroom teaching and offering courses to larger population of learners across the globe. However, all LMSs only support static courses that are not tailored for theirs learners.

The key words for a learning environment able to motivate, engage and inspire learners are adaptation and personalization. These two words are strongly connected: adaptation is based on personalization. They allow the learner to access the most appropriate, interesting and challenging learning activities, and to avoid learning material already acquired by the learner, and then not any more necessary to him.

Personalization and related adaptation are possible through adaptive learning technology (Adaptive Learning Environment). Research has invested a lot of effort in this field and several adaptive learning frameworks have been designed and implemented, like WINDS, AHA!, InterBook, APeLs. WINDS [] is an adaptive learning environment integrating an intelligent tutoring system, a computer instruction management system and a set of cooperative tools. WINDS system provides adaptability and personalization of study materials according to the learner model and history of the performed actions in a context of Virtual University of Architecture and Engineering Design. AHA! [] is an open source general-purpose adaptive hypermedia system mainly used in education domain. It supports adaptation techniques such as adaptive guiding, link annotation, link hiding and adaptive presentation. InterBook [] is a tool for authoring and delivering adaptive electronic textbooks on the World Wide Web. It supports adaptive navigation that guides the users in their hyperspace exploration. APeLS [] is a multi-model metadata driven adaptive hypermedia system that manages narrative, content and learner into different models. It is based on a rule-based engine that produces a model for personalized courses based on a narrative and the learner model.

The GRAPPLE project aims at delivering to learners a technology-enhanced learning (TEL) environment that guides them through a life-long learning experience, automatically adapting to personal preferences, prior knowledge, skills and competences, learning goals and the personal or social context in which the learning takes place. The same TEL environment can be used/accessed at home, school, and work or on the move (using mobile/handheld devices). In order to achieve this ambitious result, GRAPPLE has developed a new and improved adaptation engine, natural evolution of AHA!. GRAPPLE Adaptive Learning Engine (GALE) strives for a more flexible and powerful adaptation engine than AHA!.

2 Learning Management Systems

LMSs are based on a variety of development platforms, like Java EE based architectures, Microsoft .NET, PHP, and usually employ the use of a database back-end. Some systems are commercially developed and have non-free software licenses or restrict access to their source code. Other systems are free and open-source and frequently used. Other than the most simple, basic functionality, LMSs cater to, and focus on, different educational, administrative, and deployment requirements.

Although LMSs are very different and are built for specific purposes (corporate or academic environment), they share a quite large set of characteristics, such as:

- Manage users, roles, courses, instructors, facilities, and generate reports
- Course calendar
- Learning Path, defined as the route taken by a learner through a range of (commonly) e-learning activities, which allows them to build learner's knowledge progressively
- Student messaging and notifications
- Assessment/testing capable of handling student pre/post testing
- Display scores and transcripts
- Grading of coursework and roster processing, including waitlisting
- Web-based or blended course delivery.

The LMS is the daily tool used by thousands of learners to access the courses they have been enrolled. The added value of the integration with GRAPPLE system is to allow the learner to take advantage of an innovative system by using their LMS.

The GRAPPLE consortium has included five different LMSs, three open-source Claroline, Moodle, Sakai and two commercial learn eXact and IMC CLIX.

As described in Graf and List paper [*] and in Kuravolas[**], an evaluation of open source e-learning platforms/ LMSs was conducted with the main focus is on adaptation issues – adaptability, personalization, extensibility, and adaptivity capabilities of the open-source platforms. This evaluation involved thirty-six platforms, among them Sakai and Moodle are present. Adaptation received very little coverage in e-learning platforms. An e-learning course should not be designed in a vacuum; rather, it should match students' needs and desires as closely as possible, and adapt during course progression. The extended platform will be utilized in an operational teaching environment. Therefore, the overall functionality of the platform is as important as the adaptation capabilities, and the evaluation treats both issues. This research is focused on customizable adaptation only, which can be done without programming skills.

These LMSs adaptation criteria are [3]: (1) Adaptability – includes all facilities to customize the platform/LMS for the educational institution needs (e.g. the language or the design); (2) Personalization aspects – indicate the facilities of each individual user to customize his/her own view of the platform; (3) Extensibility – is, in principle, possible for all open source products. Nevertheless, there can be big differences. For example, a good programming style or the availability of a documented application programming interfaces are helpful; (4) Adaptivity – indicates all kinds of automatic adaptation to the individual user's needs (e.g. personal annotations of LOs or automatically adapted content).

Among the thirty-six LMSs, Moodle is the best LMS in terms of adaptation criteria, even if personalization and adaptivity are still basic.

3 Integration of GRAPPLE Adaptive Learning Environment with existing Learning Management Systems

Fig. 1 illustrates the integration of popular Learning Management Systems (LMSs) such as Sakai, Moodle, Claroline, learn eXact and CLIX into the Adaptive Learning Environment (ALE) and lists the responsibilities of an LMS, ALE, and the service oriented integration framework.

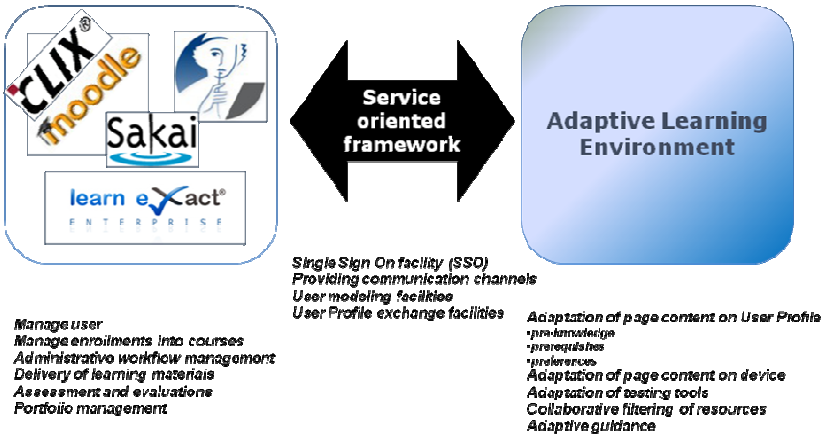

Fig. 1. Integrating popular Learning Management Systems into an Adaptive Learning Environment.

The ambitious and challenging outcome of this integration is to provide to the existing LMSs a proper adaptation and personalization of the courses the learner has been enrolled. ALE is in charge of adapting and then personalizing contents based on the user pre-knowledge, prerequisites and preferences. Since GRAPPLE solution involves a distributed system with possibly many LMSs and many users, ALE has to be able to manage all the information LMSs can provide.

The GRAPPLE Adaptive Learning Environment needs to collect all the information related to the users and their activities and the LMS is the right source of such sort of information.

Fig. 2 illustrates in more detail the two key components of ALE: (1) GRAPPLE Adaptive Learning Engine (GALE), where the content adaptation is performed; (2) GRAPPLE User Modeling Framework (GUMF), in charge of managing user model data.

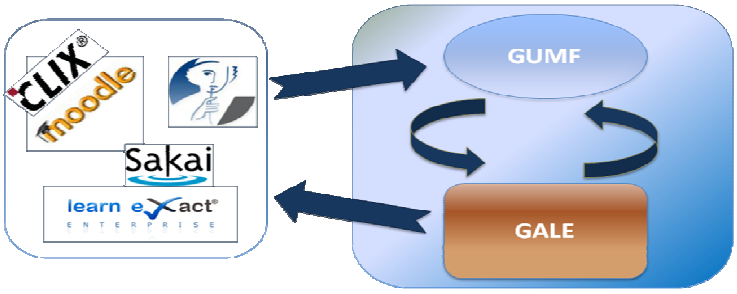

Fig. 2. Adaptive Learning Environment architecture integrated with existing Learning Management Systems.

The interaction between GALE and GUMF is bidirectional: GUMF provides all the user profile details necessary to perform adaptation based on the user knowledge, prerequisites and preferences; GALE keeps GUMF updated about the user progress.

In order to support this sort of communication, the Event Bus architecture [] was chosen for a large range of benefits: extendibility and scalability are the most relevant. Different components can be added or removed at any time.

LMS, GUMF and GALE are different components of a system and the GRAPPLE Event Bus (GEB) is the element in charge of the communication among the above components.

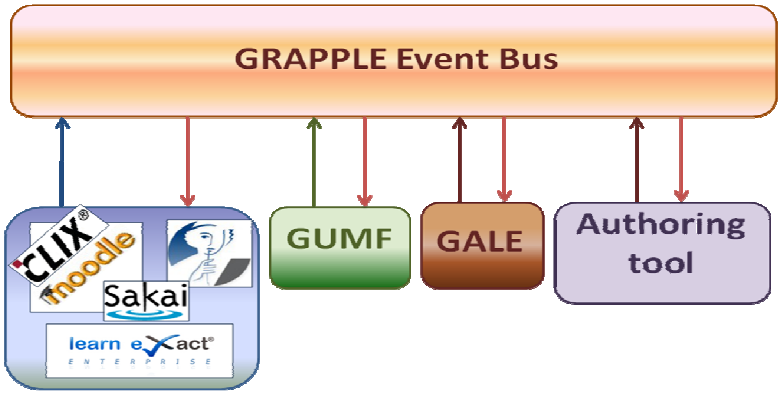

Fig. 3. Integration of existing Learning Management Systems and ALE through the GRAPPLE Event Bus.

The LMS provides a list of User Model data to the GRAPPLE User Model Framework that manages them in to provide the appropriate input to GALE. This communication is asynchronous and entirely performed by the GRAPPLE Event Bus.

In order to guarantee interoperability within the operational infrastructure and to harmonize the data flows in the GRAPPLE system, suitable data models have been defined and implemented, to be used by the LMS and ALE to exchange information through the GRAPPLE Event Bus.

In **Fig. 3** it is possible to see another component, still part of the ALE, the Authoring tool: the author is able to define the adaptive courses assigning specific storylines tailored on the user's learning progress and preferences.

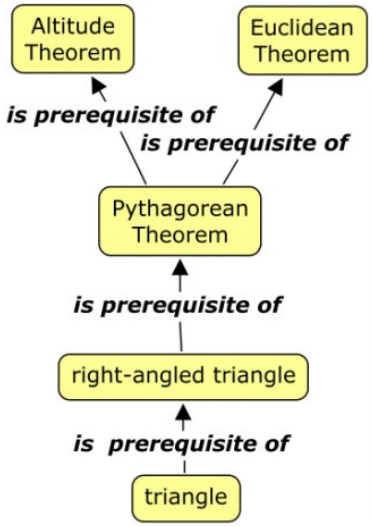

Fig. 4. Example of storyline for an adaptive course.

In order to make usable the integration of existing Learning Management Systems and ALE, it is necessary to have a Single Sign On (SSO) facility.

SSO is a mechanism whereby a single action of user authentication and authorization can permit a user to access all computers and systems where he has access permission, without the need to enter multiple passwords. SSO makes the system more usable for the authors and learners and provides easier management of access privileges to various resources.

In GRAPPLE Project we use Shibboleth for single sign-on between GRAPPLE (the GRAPPLE Event Bus is the GRAPPLE component directly involved in this) and LMSs – Moodle, Sakai, Claroline, CLIX, learn eXact.

The Shibboleth Identity Provider is able to provide and validate unique identifiers to the users. LMS and ALE initiate the requests for authentication and attributes and process incoming authentication and attribute information.

3.1 GALE: Grapple Adaptive Learning Engine

The Grapple Adaptive Learning Engine serves content through regular http requests. The resulting document is adapted to the user's needs on the fly. To perform this adaptation GALE requires input from the author of an adaptive course. The author defines concepts, attaches resources to concepts, defines actions to be taken (e.g. user model updates), when certain events occur (e.g. page access), and more. The author also defines the adaptation that needs to occur based on a particular user model. GALE allows authors to use powerful Java code to achieve adaptation. We are able to shield authors from too much complexity through templating techniques.

Because adaptation is done on the fly, GALE requires the user model to be readily available. Furthermore some parts of a GALE user model are not interesting to the outside world. Their only function is internal to GALE (e.g. how often was a concept visited). We allow the author to decide which specific GALE user model variables are to be made public. Likewise the author can indicate which user model variables in GALE are the local representation of public user model information.

Public user model information is communicated through GRAPPLE statements. A single piece of information is identified by its URI. The GUMF and GALE communicate with each other through the GRAPPLE Event Bus by means of these GRAPPLE statements.

When an author decides that some GALE user model variable is public, he attaches a property named 'public' to that variable in GALE. The content of the property is the URI of the GRAPPLE statement to produce or obtain. A public GALE variable can be 'authorative', meaning GALE is the authority for its value. Any change to its value from within GALE is made public (a 'Tell' request is send to GUMF). If the variable is not defined as 'authorative', its value is retrieved (an 'Ask' request is send to GUMF).

The GRAPPLE Event Bus is asynchronous. This allows a lot of the work involved in sending and retrieving public user model information to happen in the background. However, GALE does not wait for this public user information. It continues adaptation as if all values were instantly available (default values are used whenever this is not the case). The page is returned to user when adaptation is finished. GALE uses Ajax technology to update the page dynamically, as new information about the user becomes available.

3.2 GUMF: Grapple User Modeling Framework

The Grapple User Modeling Framework (GUMF) provides a central point of access similarly to the *generic user modeling systems* proposed in [1]. It enables client applications to query for information about users and allows for storage of user data. GUMF contextualizes information that is stored by the clients, i.e. it enriches the information with metadata describing provenance, temporal and spatial validity, etc. and it connects the information with data from other sources spread across the Web as well as with GUMF-internal sources. The contextualization of data stored by some client application is not only beneficial for improving the user modeling capabilities

of GUMF, but especially also for other client applications that are allowed to access the data as they get clear insights in which context some piece of information was generated. Atomic pieces of information in GUMF are called Grapple statements [2] and are built on the notion of reified *subject-predicate-object* statements that are enriched with contextual metadata. Those Grapple statements, which are a further development of statements adhering to the General User Model Ontology [3] (GUMO) and UserRDF [4], constitute the common exchange format for clients that communicate with GUMF. Grapple statements can be considered as containers that allow for the integration of arbitrary domain ontologies and therwith support popular user profile vocabularies such as FOAF [5], SIOC [6], and OpenSocial/RDF [7]. The user modeling capabilities of GUMF are fully adaptable to the requirements of the particular client applications. The administrator of a client application can adjust user profile reasoning functionalities, define access control rules and specify schema mappings so that the client applications can operate on their preferred domain ontologies. In addition to the rich set of user modeling functionality provided by the core GUMF distribution, developers are enabled to extend GUMF's user modeling capabilities by means of plug-ins. For the development of extensions, it is possible to make use of already existing GUMF features and extensions.

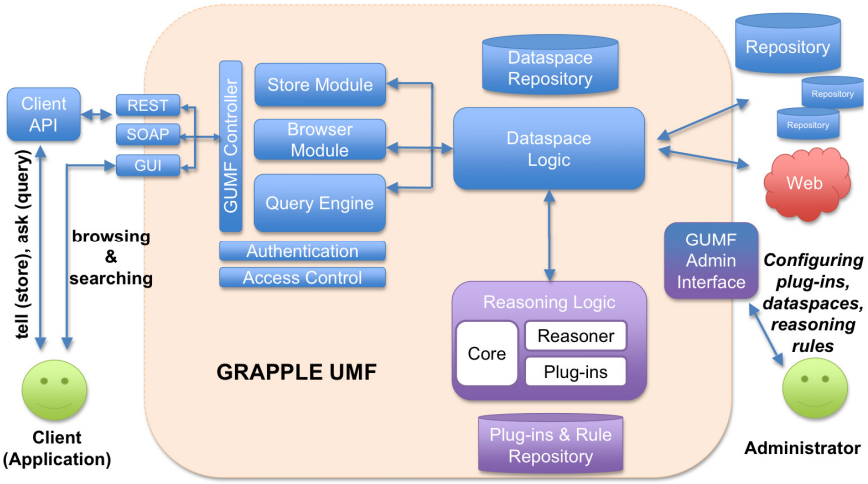

Fig. 5. Architecture of the Grapple User Modeling Framework (GUMF).

Fig. 5 shows the architecture of GUMF. The elements at the top provide the essential, generic functionality of the framework; elements part at the bottom right provide generic as well as domain-specific *reasoning logic*. Client applications can access GUMF either via a RESTful or SOAP-based API. Further, there is a *Java Client API* that facilitates development of GUMF client applications. Client applications mainly approach GUMF to store user information (handled by the *Store Module*) or to query for information (handled by *Query Engine*) whereby Grapple statements [2] are used to model user profile information. GUMF currently supports SPARQL [8] and SeRQL [9] queries as well as a pattern-based query language that

exploits the Grapple statement structure to specify what kind of statements should be returned by GUMF. Authorized client requests are answered by GUMF's *Dataspace Logic*. Dataspaces are equipped with data storage repositories that either reside at the GUMF server or are distributed across the Web (possibly maintained by the client application itself), and with (reasoning) plug-ins that further enrich the data available in the repositories. The *Administrator* of a GUMF client application can configure dataspaces and plug-ins via the *GUMF Admin Interface*. Activating or deactivating plug-ins directly influences the behavior of dataspaces. Further, administrators can adjust the plug-ins and reasoning rules to their needs. For example, we developed a plug-in that gathers user profile information from Facebook and maps---with support of Silk [10] – the profile to a format preferred by the client application administrator (e.g., FOAF [5] or OpenSocial [7]).

Inspired by Web 2.0 practices, a key principle of GUMF is that dataspaces can be shared across different client applications. Therefore, clients can *subscribe* to other dataspaces, given that they are granted approval by the administrator of the dataspace. When subscribed to a dataspace, the client is allowed to query it. However, it might still not be allowed to access all statements that are made available via the dataspace, as fine-grained access control functionality can be embedded in the dataspaces as well.

4 Conclusions

This paper highlights how the existing LMSs are not able to provide to the final students learning activities tailored to their needs, even if many attempts have been tried in the latest years.
In this paper we explained how in GRAPPLE we have proceeded to satisfy this issue by involving an engine able to deploy adaptive courses, a user model framework able to provide the proper user data and an authoring tool that allows creating courses.

Acknowledgments. The work presented in this paper has been partially sponsored by the EU FP7 STREP Project GRAPPLE.

References

1. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195--197 (1981)
1. Kobsa, A.: Generic user modeling systems. User Modeling and User-Adapted Interaction 11(1-2) (2001) 49–63
2. Abel, F., Heckmann, D., Herder, E., Hidders, J., Houben, G.J., Leonardi, E., van der Sluijs, K.: Definition of an appropriate User Profile format. Technical report, Grapple Project, EU FP7, Reference 215434 (2009)
3. Heckmann, D., Schwartz, T., Brandherm, B., Schmitz, M., von Wilamowitz-Moellendor□, M.: GUMO - The General User Model Ontology. In: Proc. of 10th Int. Conf. on User Modeling, Edinburgh, UK (2005) 428–432

4. Abel, F., Henze, N., Krause, D., Plappert, D.: User modeling and user profile exchange for Semantic Web applications. In: 16th Workshop on Adaptivity and User Modeling in Interactive Systems, Wuerzburg, Germany. (2008)
5. Brickley, D., Miller, L.: FOAF Vocabulary Specification 0.91. Namespace document, FOAF Project, http://xmlns.com/foaf/0.1/ (November 2007)
6. Bojars, U., Breslin, J.G.: SIOC Core Ontology Specification. Namespace document, DERI, NUI Galway, http://rdfs.org/sioc/spec/ (January 2009)
7. Nowack, B.: OpenSocial/RDF. Namespace document, http://web-semantics.org/ns/opensocial (December 2008)
8. Prud'hommeaux, E., Seaborne, A.: SPARQL Query Language for RDF. W3c recommendation, W3C (January 2008)
9. Broekstra, J., Kampman, A.: An RDF query and transformation language. In Staab, S., Stuckenschmidt, H., eds.: Semantic Web and Peer-to-Peer, Springer Verlag (March 2006) 23–39
10. Volz, J., Bizer, C., Gaedke, M., Kobilarov, G.: Silk – A Link Discovery Framework for the Web of Data. In Proc. of 2nd Workshop about Linked Data on the Web (LDOW) in conjunction with the 18th International World Wide Web Conference (WWW), Madrid, Spain (April 2009)