GRAPPLE: Personalization and Adaptation in Learning Management Systems

Paul De Bruin, David Smith, Kees van der Sluijs, Alexandra I. Crisostomi, Maurice Hendriks, and many other members of the GRAPPLE Research Team

Abstract: Learning Management Systems (LMS) are used in many schools, colleges, and universities. This paper focuses on supporting the learning process at a fairly global level of courses and tests. Recent additions focus on interaction (discussion forums, chat rooms, and virtual). GRAPPLE tackles an important issue: understanding the adaptive delivery of course material into the supported learning process. The project aims to support the adaptive learning system by providing a more integrated and personalized learning experience for students.

Introduction

Support for learning is highly dependent on the management and the process. Many educational institutions use Learning Management Systems (LMS) to manage and track the progress of students through the curriculum. The purpose of the project is to develop a system that is highly flexible and can be adapted to the needs of individual students. The project involves the development of a tool that can be used to support the learning process. The project aims to provide a more personalized and adaptive learning experience for students.

Adaptive technology-enhanced learning (or adaptive TEL) has been a major application area for adaptive hypermedia. The seminar by Peter Brusilovsky (Brusilovsky, 1998, Brusilovsky, 2001) has been a source of inspiration for many researchers into personalized information access. A few years many new methods and technologies were developed, as summarised in (Kasivisv, et al., 2000, 2001). Many special-purpose tools were developed and also one general-purpose adaptive system: AHA! (De Bruin et al., 2000, 2001), that has been used by researchers and educators in many countries all over the world. However, to date the use of adaptive technology in learning applications has remained rather limited because it was not integrated into the learning management system used by the schools and universities.

GRAPPLE bundles the expertise of researchers from 15 universities, research institutes and companies, including the creators of the adaptive systems AHA! (De Bruin et al., 2000), KBS-Hyperbrok (Hencee et al., 1999), and many other members of the GRAPPLE Research Team.

1 Paul De Bruin, David Smith, and Kees van der Sluijs work at the Eindhoven University of Technology, the Netherlands.
2 Alexandra I. Crisostomi and Maurice Hendriks work at the University of Warwick, UK.
3 Some noteworthy examples are the aLeCo (Learning to code) project from the University of Bielefeld (Berthelot, 2006, 2007), an automatic text document from Kona University (Lee et al., 2005), and a programming course from the Slovak University of Technology (Bieliková, et al., 2005). We are also aware of ongoing work in Brazil, Colombia, and South Africa.

RATH (Hockenmayer et al., 1998), ApNLS (Cordani et al., 2002) and WINDS (Kravetc et al., 2004), of authoring systems and adaptation languages (Crissato et al., 2008), of user modeling languages and services, including UML (Heckman et al., 2003, Heckman et al., 2005) and the work of (Van der Sluijs et al., 2006), of experts in learning standards (e.g. the Open University Nederland and Acompio Spain), of developers and contributors to LMS including Moodle, Claroline and Sakai, of developers of industrial TEL applications (Aneos, Gislab Labs and IMC Information Multimedia Communication AG). The goal of GRAPPLE is to have the adaptive learning environment (or ALE) become a "standard" component of the LMS that the thousands of universities in the world can use these LMSs automatically have access to the ALE.

In this paper we first describe the overall GRAPPLE architecture, emphasizing how the ALE and LMS can work together to offer an integrated adaptive learning solution. We then describe the processing of creating adaptive course material and using it from within the LMS. We describe the conceptual authoring tools and the different ways to create content. The tools are going through a continuous process of improvement based on input from evaluation questionnaires given to the early adopters of the GRAPPLE technology.

Communication between an ALE and LMS in GRAPPLE

The standard way to set up an LMS environment involves creating user accounts (for teachers and learners) on the LMS. Applications other than the LMS have no access to these accounts. One possible exception is that some LMSs have a provision for accessing another application from within the LMS and passing the user id. At the TUI we have used this to give students with an account on our Sakai LMS access to the adaptive course test for our course 2ID5 on hypermedia. Unfortunately the "Link Tool" that is used (at TUI within Sakai) to provide this access is insecure. Furthermore the access is one-way only: from Sakai students can access the adaptive course test but the interaction with the course test cannot be communicated back to Sakai. For instance to have the progress of the student recorded in his/her grade book. Also, tests or exams the student may perform in the LMS cannot influence the adaptive learning done by the course test as only the student id is passed on.

In GRAPPLE we have arranged for communication between LMSs and instances of the ALE through the GRAPPLE User Model Framework (GUMF). An ALE can tell GUMF that a student has performed a test (and what the grade was) and GUMF can provide that information to the ALE. The adaptation can then take the test results into account. Likewise, the ALE can tell GUMF about the progress of the student and this can be communicated to the LMS, perhaps to determine when to enable access to a test or assignment. An infrastructure using an asynchronous event bus is used to facilitate the communication between all GRAPPLE components.

GRAPPLE is intended to support life-long learning, meaning that GUMF must be able to share information about a learner with different LMSs. When the learner moves to a different school, university or company, the information a learner decides to disclose must be accessible for a new LMS. Therefore GRAPPLE is designed in a system-independent way. This can be done through the project-specific "GRAPPLE statement" but also through the IMS Learner Information Packaging (LIP) standard. Within GRAPPLE five different LMSs can communicate with each other and with the GRAPPLE ALE (GALIE). These LMSs are the open source LMS Moodle, Sakai and Claroline, and Eate from Gislab Labs and the GRAPPLE ALE (GALE). GALE is used to deliver the course material and the ALE is used to provide the links to the LMS.

As can be seen from Figure 1 all GRAPPLE components communicate with each other through a shared GRAPPLE Event Bus (GEB). This is an asynchronous communication bus, meaning that components can send data or requests to another component through the bus and can listen for answers but cannot just wait for an answer to their request. As a consequence, components should be designed to maintain their normal operation without the answers to their requests; the answers will arrive asynchronously. This is an important issue for the adaptive delivery of course material as GALE may not always be immediately notified when a student completes a test on the LMS, and may thus not immediately perform adaptation according to that changed user model. GALE has an internal user model store to enable it to offer adaptability based on interactions with GALE almost instantly.

The GRAPPLE environment is distributed by design. The authoring Tool (GAT), the Adaptive Learning Environment (GALIE), the User Model Framework (GUMF), the Event Bus (GEB), the LMSs and additional tools such as special visualization tools (GVIZ) may all be running on different servers anywhere on the Internet (and in the project setup they actually do run on servers in different countries). In order for users to be known by

http://www.imsproject.org
http://www.gislab.com
http://www.im-c.com
Creating the conceptual adaptation model for a course

Figure 3 shows a partial domain model in the DM editor of GAT. It shows an example application called “Milkyway” that we also use to illustrate other parts of the authoring process in this paper (and in other presentations). In the figure we see the Sun, planets, and moons and relationships like “isMoonOf”, connecting moons with their planet, and “isTheOf” connecting moons with the abstract concept “Moon” and connecting planets with the abstract concept “Planet”. Some parts of the model in this figure are incomplete: “isPlanetOf” relationships between planets and the Sun are still missing as are some moons. Each concept in DM can have an arbitrary number of properties with values and resources (references to files) with properties and values. The properties can be used by the adaptation engine, for instance to select a resource to show depending on some user model properties.

3. Concepts from DM are connected using CRTs to form the Conceptual Adaptation Model (or CAM®) (Hendrix et al., 2006). In past research on AHA! (De Bra et al., 2006) and in other systems as well the CRTs typically were either unary or binary relationships between single concepts. In GRAPPLE a concept relationship can connect an arbitrary number of concepts or even sets of concepts. For this it uses sockets each containing one or more concepts. Figure 4 shows a screen shot of a first part of the CAM for the Milkyway example. Allowing sockets with multiple concepts allows for the creation of CRTs with associated adaptation rules that take all concepts of a socket into account. Also, it greatly reduced the number of relationship (instances) that need to be created.

---


4 SDM and VR-DM in Figure 2 refer to simulations and virtual reality extensions, not described in this paper.
4. When the CAM has been completed the author selects the "Deploy" function (from the File menu). This sends the CAM to the GALE engine through what Figure 2 calls the "Engine-Specific Compiler". The CAM includes the DM (or DMS) used in the CAM and also the definition of the CRTs used in the CAM. The compiler creates GALE-concepts and relationships from the DM description and translates template adaptation code specified in the CRTs into default code for initializing user model variables and event code for associating user-initiated events such as a page access or system-generated events such as a user model update with code to generate user model updates (possibly defining adaptation behavior). We illustrate the adaptation code with an example below. Please note that an author who uses predefined CRTs does not need to write or even see such code, and that the majority of authors will only use the predefined CRTs.

In the Milkyway example we indicated that "Earth" is a prerequisite for "Moon". The prerequisite CRT has the following adaptation code:

```plaintext
$target{ if ($suitability and $suitability<"Earth" and $suitability>"Moon") }
```

This code specifies that the suitability attribute of each concept in the "target" socket should be initialized not with a plain value but by executing (hence "if") the code $suitability<"Earth" and $suitability>"Moon". This bit of code is replicated for each concept in the "source" socket and if there are several such concepts the expressions are combined using the logical and operator ("&&"). For our example prerequisite this means that the suitability attribute of concept "Moon" will be initialized with the value that is the result of evaluating the expression Earth<"knowledge">Moons.

One remaining aspect we have not covered is that of the "location" of user model information. In the CRT tool an author defines that a CRT uses some user model attributes (for concepts appearing in a specific socket). Normally the (run-time) values of these attributes of concepts are only known by the GALE engine with which the learner interacts directly. The values are used for generating adaptation, for instance based on the learner’s knowledge of some concepts, in combination with the existence of prerequisite relationships. However, the author may indicate that some attributes are public. When an attribute is persistent (i.e. stored and not calculated on the fly like the suitability in the example above) and public, GALE will communicate all changes of this attribute to GUMP, making the up to date value available to other GRAPPLE components (like an LMS or perhaps another GALE server serving another course). When an attribute is not persistent and public, GALE will request the value from GUMP and will register a listener on the event bus to stay informed of updates of that attribute. Other than having this specification (public or not, persistent or not) the author need not be concerned with the communication with GUMP when defining the CAM or creating the content of a course.

Creating content for a GRAPPLE course

Each concept in the DM of a course can be associated with one or more resources. Typically a resource is a file (an HTML page) that forms a possible presentation of the concept. Having multiple resources allows for the automatic selection of one of several alternative presentations, for instance a version for a beginner and a more advanced learner. The adaptation rules for selecting the appropriate version can be specified at the conceptual level using the CRT tool.

In an online course we expect consistency in the presentation format. In GRAPPLE there is an additional constraint that the presentation must also fit "within" the LMS presentation structure. To end GALE always generates a single HTML page (at least if the input is also HTML), using tables for different parts of its presentation. Achieving presentation consistency is done by means of two complementary techniques: layout definitions and page templates.

1. In GALE concepts can extend other concepts, meaning that they inherit attributes from these other concepts. An extend relation between concepts is used for this. The standard "admin" application contains a concept "_layout" that has a single attribute layout with the following default value:

```html
<struct cols="20%;"><!DOCTYPE="static-tree-view"> </content/></struct>
```

Every concept that extends the "_layout" concept will be presented using a navigation menu for the course on the left (in a column taking up 20% of the width) and the content page on the right (in the remaining 80%). GALE has a number of predefined "views" to show part of the DM of an application. In the standard
layout the static-tree-view is used, which shows an accordion menu where the top-level concepts are shown and the part of the concept hierarchy containing the current concept is expanded (and the remainder of the concept hierarchy is collapsed).

2. For the pages themselves, there are also two main approaches: creating each page "from scratch" (writing pages or importing "ready-made" pages from a content authoring system, such as the MOT adaptive hypermedia authoring system (Foss & Cristea, 2010)) or using template pages. The hypermedia course 2ED51 at the TUEI describes many different topics. The structure of different pages or parts of the course is different. Each page in this course is therefore authored separately. The only common aspects of the presentation are the common layout (and of that also two versions are used in this example). The left part of Figure 5 illustrates this authoring approach. The Milkyway example presents some information about celestial objects. The actual content is copied from Wikipedia. The presentation of similar objects is identical. Figure 6 shows cropped screen dumps from the pages about Jupiter and Saturn. They start with a title, then describe the position of the object in the solar system ("Is Planet of: Sun"), then gives an image, followed by textual information, and finally there is a list of moons of the planet. It is reasonable to expect that any change to the presentation an author may come up with should be applied to the pages about each planet. Therefore, we associate the concepts with a common resource which is the template page. Every item that is "planet-dependent" is either the value of a property of the concept or is contained in a file of which the URL is a property of the concept. GALE offers some xml tags that can be used in pages (using both the xstmi and gate namespaces) to include such information:

- The `<gate:variable>` tag is used to present the value of a user model attribute or the result of an expression over (domain- and) user model. Two examples:

```xml
<gate:variable name="gale://gale.tue.nl/personalName" />
```

will be replaced by the name of the user. (personal is a pseudo-concept that represents the user)

```xml
<gate:variable expr="@keyup"/>
```

will be replaced by the title property of the current concept.

- The `<gate:object>` tag is used to include a file in place of the object. The file must contain a valid xml fragment. The html `<img>` tag can also be used to include a file which must be an image. In order to determine the source for the object or image from the domain or user model the `<gate:attr-variable>` tag can be used. This is a "trick" because inside an xml tag no xml tags can be used.

```xml
<img width="300"><gate:attr-variable name="src" expr="/51.jpg/image"/>
```

will load the image of which the URL is obtained by evaluating the expression that retrieves the value of the attribute `image` of the current concept.

![Diagram](image)

**Figure 5:** Authoring pages separately (left) or using template pages (right)

**Conclusions and Future Work**

The GRAPPLE project attempts to bring adaptive course delivery to the masses using a three-pronged approach:

1. it incorporates the adaptive learning environment (ALE) in popular learning management systems (LMSs),
2. it offers easy-to-use graphical authoring tools for the conceptual adaptation model (CAM) and
3. it offers a powerful adaption engine that allows more popular adaptation techniques for both content and navigation structures.

Although the project and the software development are still ongoing, training events have already been held in academic and industrial environments. Each event is followed by an evaluation (questionnaire) from which suggestions for further improvement are extracted. Students in a course on adaptive systems are currently using the authoring tools (GAT) and adaption engine (GALE) in group assignments in which they are creating adaptive course material. GALE is also already in use for course delivery at the TUEI and has proven to have high performance and reliability.

Although GAT+GALE are currently actively being used and evaluated, the two-way communication between GALE and different LMSs is still undergoing testing. Also, the use of Shibboleth for the single sign-on facility is still being tested.
Further developments in the GRAPPLE project, before, during and after HD-MEDIA 2010, can be followed at www.grapple-project.org.

Acknowledgement
This work has been performed in the framework of the IST project IST-2007-215434 GRAPPLE which is partly funded by the European Union. The authors would also like to acknowledge the contributions of their numerous colleagues from all 15 GRAPPLE project partners.

References


