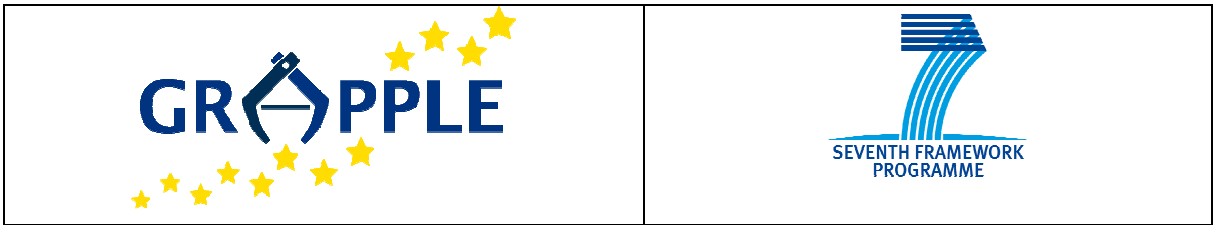

GRAPPLE

D7.4b Version: 1.0

Operational infrastructure - second prototype

Document Type	Deliverable
Editor(s):	Lucia Oneto (GILABS), Alessandro Mazzetti (GILABS)
Author(s):	Lucia Oneto (GILABS), Luca Mazzola (USI), Kees van der Sluijs (TUE)
Reviewer(s):	WP1-6
Work Package:	WP7
Due Date:	31-10-2009
Version:	1.0
Version Date:	22-3-2010
Total number of pages:	9

Abstract: The second prototype is accompanied by this report that explains how to use it where existing LMSs and GRAPPLE framework are integrated.

Keyword list: Learning Management Systems, Adaptive Learning Environment, integration

Summary

The GRAPPLE system is a component-based service oriented system: the major functionalities are split into modules or components, which in turn are split into services. The purpose of this deliverable is to document and accompany the second prototype of the GRAPPLE project.

The functional architecture of the GRAPPLE system and its integration with existing LMSs in academic and industrial settings is documented in D7.1b - Updated specification of the operational infrastructure that describes the current separation of the whole system into independent components interfaced to external LMSs, and the infrastructure that compose the GRAPPLE Framework.

Authors

Person	Email	Partner code
Lucia Oneto	l.oneto@giuntilabs.com	GILABS
Luca Mazzola	luca.mazzola@usi.ch	USI
Kees van der Sluijs	k.a.m.sluijs@tue.nl	TUE

Table of Contents

Tables and Figures

List of Figures

List of Acronyms and Abbreviations

ALE	Adaptive Learning Environment
GCC	GRAPPLE Conversion Component
GRAPPLE	Generic Responsive Adaptive Personalized Learning Environment
GUMF	GRAPPLE User Modeling Framework
IMS	Instructional Management System
IMS LIP	IMS Learner Information Package
LMS	Learning Management System
EPMB	Executive Project Management Board
GEB	GRAPPLE Event Bus
GALE	GRAPPLE Adaptive Learning Engine
GVIS	GRAPPLE Visualization
GAT	GRAPPLE Authoring Tool
GDA	GRAPPLE Device Adaptation
DM	Domain Model
UM	User Modeling
CRT	Concept Relationship Type
CAM	Conceptual Adaptation Model

1 Introduction

1.1 Task and Deliverable Description

T 7.4 Intermediate operational infrastructure implementations: (GILABS, USI, UCL, UC, IMC, ATOS, TUe, LUH, DFKI)

The implementation will be accomplished in different phases to support the delivery of two intermediate prototype systems (before the final release). Each phase will include the design/refinement and implementation of interfaces amongst components and their integration into a coherent and consistent system. Incrementally improved releases of the GRAPPLE components and services will be integrated in corresponding GRAPPLE prototypes and evaluated according to the criteria and protocol defined in task 7.3 (from the technical perspective) and WP8 (from a wider and more pedagogy-oriented perspective).

D7.4a Operational infrastructure - first prototype (GILABS, M12)

This will be the first prototype of the GRAPPLE system, where a subset of the identified functionalities and features will be made available. Suitable accompanying documentation will be produced.

D7.4a Operational infrastructure - second prototype (GILABS, M21)

Within the incremental development approach adopted in the project, this will be the second prototype of the GRAPPLE system. With respect to the first version of the system, this will offer more and better integrated functionalities. Also in this case suitable accompanying documentation will be produced.

2 GRAPPLE Operational Infrastructure Overview

The first prototype described in D7.1a was the first step towards the GRAPPLE framework able to be integrated with existing LMSs. In the first year of the project only two modules were available: GALE, the GRAPPLE Core Engine, and the LMSs belonging to the consortium.

The second prototype involves a considerable improvement in comparison with the previous version: the other Work Packages (WPs) have developed new GRAPPLE modules and they need a solid and stable infrastructure able to integrate them.

Figure 1 displays the GRAPPLE components and the communication among them. It is possible to identify a key element, the GRAPPLE Event Bus that handles the connections among the LMS, GUMF, GALE, the GRAPPLE Core Engine, GAT and the GVIS components, and Shibboleth as technological choice for web single sign-on across or within organizational boundaries.

GCC provides an interface for LMS to the GEB. It is therefore a wrapper that converts LMS specific functions and data structures into information that can be handled by the GRAPPLE infrastructure. This justifies its collocation in Figure 1: the single GRAPPLE Conversion Component manages and maps the LMS specific parameters to standard format (IMS LIP), as specified in D7.2b. Then it is necessary to have a GCC for any LMS integrated to GRAPPLE.

GDA component is responsible of the content adaptation based on the type of device that is used by a learner to access to the adaptive course. This component is a module located in the GALE, the GRAPPLE core engine in charge of delivering the adaptive courses.

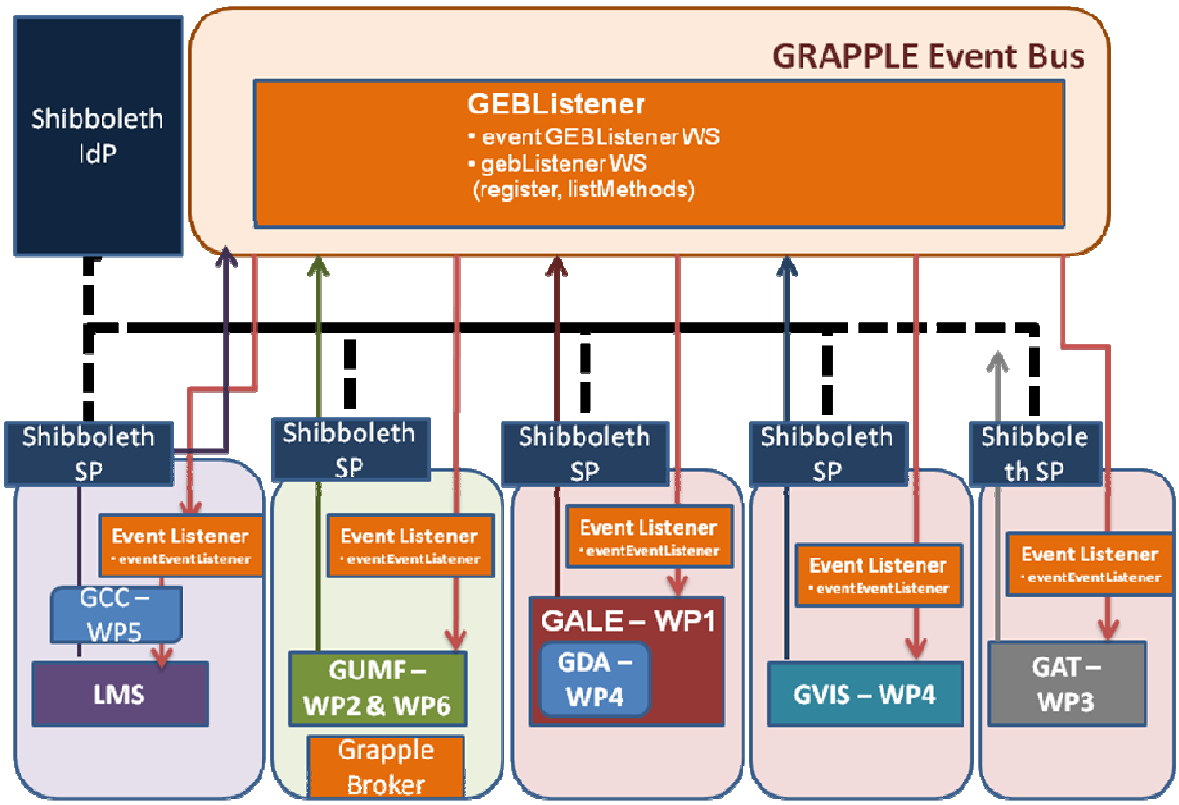

Figure 1: GRAPPLE second prototype architecture.

All components are connected through the GRAPPLE Event Bus (GEB). Each component may send requests to the GEB, where other components may listen to these events and handle them. This may result in a reply sent to/through the GRAPPLE Event Bus as well. In specific, (1) all the GRAPPLE components send just events and other ones receive these events if they listen and (2) any GRAPPLE component can send requests to the GEB and only the addressed component responds.

Figure 1 illustrates the communication among the main GRAPPLE components: even if they are not present, any component can provide a set of web services.

3 Demo Scenario – LMS and GRAPPLE framework integration

For presenting the core interaction between the system services a scenario-driven description approach has been selected: the chosen scenarios have been proposed for the second prototype and they involve the following GRAPPLE components:

- LMS (Learning Management System) in charge of delivering the non-adaptive courses and of providing a container for the adaptive courses.
- GALE (GRAPPLE Adaptive Learning Engine) as GRAPPLE Core Engine in charge of delivering the adaptive courses. GALE includes its own authoring component and user model.
- GAT (GRAPPLE Authoring Tool) in charge of the authoring part. It is composed by three elements, DM (Domain Model), CRT (Concept Relationship Type) and CAM (Conceptual Adaptation Model) tools.
- GUMF (GRAPPLE User Modeling Framework) provides the functionality for storing, sharing and querying user model attributes, as provided by the LMS and GALE. Customizable reasoners and mappers are available for converting user model data upon request.
- GVIS (GRAPPLE Visualization) is in charge of providing meaningful representations of the User Model data enriched with data from GALE, LMS and from the full learning environment in order to support learners during self-reflection and to offer an informative tool to tutors and teachers.
- GCC (GRAPPLE Conversion Components) provide the interoperability layer between the single existing LMS and the GRAPPLE components.

- GDA (GRAPPLE Device Adaptation) is in charge of the content adaptation in function of the device characteristics.

This session describes how the second GRAPPLE prototype can be used in three scenarios where authors/tutors and students can be involved.

The following scenarios describes end-to-end the complete cycle of use of the first GRAPPLE prototype.

3.1.1 Scenario 1 – Paul and the Solar System application

Author and Teacher Paul authors an application (adaptive course) about the Solar System in GRAPPLE with the GRAPPLE authoring tools and integrates an existing quiz from an LMS, which he has authored some time ago. The LMS shows the progress of the learners.

1. LMS

At the GRAPPLE setup the LMS administrator Carl decides which variables can be available for policy reasons. By default all the variables made available by Carl are private.

Now, Paul defines which UM variables available in the LMS can be public, such as the quiz score. Any time Paul can change the UM variables status to public or private.

Paul uses the LMS in one of the LMSs to offer a quiz, sun.quizScore, to the user about the Sun concepts. The quiz has 5 multiple choice questions, and the result of the test by a user will be a number between 0-5. Paul indicates that the result of the quiz is externally available.

In order to guarantee interoperability between GRAPPLE system and different LMSs, when possible, the most of the LMS variables are mapped to a common data format valid for all the GRAPPLE system. However this can be not possible for all the variables.

2. DM

Paul creates a new domain in the domain tool. It contains concepts like “sun”, “earth”, “moon”, and relationships like “hasplanet” and “hasmoon”.

3. CRT

Paul uses the CRT tool to create the simplistic Paul_prereq CRT that specifies that:

“in order for a user to see information about concept X, the user needs enough knowledge in the user model about concept Y and enough knowledge in the user model about concept Z”, and an instantiation of this rule will mean that “all links to a ‘page’ about concept X will get a condition (about concept Y and Z) that if the condition is true (so the condition is satisfied) these links are highlighted, otherwise the links are hidden.

Through the CRT Paul indicates the expected level of proficiency for the UM variable knowledge will be 65%.This expected level will be used in visualizations (GVIS) to represent the learning objectives of students (related to that particular concept).Paul can use existing User Model variables in his crt, or create new ones. As he prefers using existing conventions and names, he decides to look what UM variables are already present. For this purpose the CRT tool has a simple UM query interface. The query interface can query for all variables in use in any adaptive course (that is uniquely associated to a CAM) by anyone, or be more specific. Paul could specify a specific CAM (adaptive course), or a specific concept. The result of the query is shown as a list of UM variables and the type and range values (if available) can be seen in the attributes area where the UM variables are defined.

4. CRT

In order to indicate, the expected end state(s) of a adaptive course Paul creates a CRT (*end-of-course-crt*) that allows to establish (via placeholders) which concepts are involved in the end state and he can indicate the expected level of proficiency that the students should have reached at the end state.

5. CAM

In the CAM tool, Paul indicates that the Paul_prereq needs to be applied to the concept ‘earth’. The prereq refers to the concept ‘sun’, with the predicates ‘externalKnowledge’ and ‘knowledge’ and a level of 100%.

This means that Paul wants to give access to a page about concept “earth” if the user has acquired internal knowledge about the Sun. In addition, Paul also wants to make sure that the user not only knows about the sun via navigating through the adaptive application, but also via filling in a test in one of the LMSs CLIX, CLAROLINE, SAKAI or learn eXact (or some other LMS). The UM variable “sun.externalKnowledge” represents having passed one of these tests in one of the LMSs: it can be mapped to a Boolean value.

In the CAM tool, Paul drags and drops the main concepts into the main-concepts-crt, as he wants to enable the visual representations of the knowledge of concepts of the adaptive course. This will ensure that the selected concepts will be used in the visualization to represent the learning objectives of the students. When students have to choose between 2 assignment topics, Paul has to indicate two slightly different possible end states. The advantage of indicating an end state is that the visualization will be able to derive and visualize the learning goals from this. In addition to this it makes it clearer for Paul where his adaptive course can end. The consistency check in the CAM will check that there are no unreachable states in it, as far as static analysis goes, so this will help Paul to make sure students can actually reach one of his intended end states and therefore achieve the learning goals.

6. GUMF mappings

Paul has created a test in the LMS called SolarSystemInterest_test. This test tests students on their interests. It aims to find out whether they are more interested in elementary particle physics, or in astronomy. This test is different from a conventional test where the result ultimately is pass or fail. Paul could write adaptation rules for this behaviour directly in a CRT. However with possible sharing in mind, Paul decides to map the results of his test, which will be a number between 0 and 10, to the UM variable interest, which can either take the value "*particle_physics*" or "*astronomy*".

In order to do this, there is a GUMF mapping tool that can be started from GAT. He opens the GUMF mapping tool, locates the test and specifies the mapping.

7. CRT

Paul also wants to show the progress of the user in the adaptive course in the LMS. Therefore in the CRT, Paul indicates which UM variables are public, i.e. shared with the GUMF. Paul selects the knowledge attributes of the subset of concepts in the adaptive course.

8. GALE

Once the application has been defined in the CAM, Paul makes it available to GALE. The process will be triggered by Paul in the CAM interface.

9. LMS

Paul makes the application Solar System available from the LMS and enrols students following the normal procedure.

10. GVIS

Paul logs on the LMS and enables a graphical widget that provides a compact indicator of the learner's progress in the Solar System application (for this particular course). Paul wants to investigate the adaptive course status, and review the progress of learners over adaptive course. He notices that the **overall indicator** reports interesting information on the global status of the class. When the instructor clicks on it, a different view is opened, and reaches a **detailed view** where a graphical representation of the **knowledge level**, and the **expected target level**, is presented for every concept of the adaptive course. In this way, the instructor can notice in which concepts the students are performing low and take appropriate actions.

3.1.2 Scenario 2 – Kees and the Space Physics application

Author and Teacher Kees authors an application (adaptive course) about Space Physics in GRAPPLE with the GRAPPLE authoring tools. Kees knows Paul and his Solar System application. As Kees knows that some students that did Paul's adaptive course on the Solar System, will also do the Space Physics adaptive course, Kees wants to prevent that users that already know concepts that were already discussed in the Solar System adaptive course have to redo them again.

1. DM

The domain of the Space Physics application contains concepts like "planet" and "star".

Kees can access to the Solar System application and use some existing concepts such as "sun" and "earth".

2. CRT

Now Kees wants to update the knowledge for these concepts by using the information from Paul's concepts of "earth" and "sun".

Kees can get the list of UM variables already available from the GUMF and he can decide to use some of them or to create new UM variables. Then Kees can select and use some UM variables already defined and available in the GUMF (made available by Paul to the other authors in Step 7).

Kees realizes he has to work with the placeholder sockets, in order to ensure the reusability of his CRT. With the sockets X and Y, Kees can choose to use directly Y.knowledge variable or to create Y.knowledge(ext) that will be filled by a UM mapping. Kees decides to create a rule that when X.knowledge is larger than 80%, Y.knowledge is increased with 10%.

Kees defines the CRT Kees_prereq that specifies that:

"if the learner has knowledge of the concept Y and/or Z, an instantiation of this rule will mean that "all links to a 'page' about concept X will get a condition (about concept Y and Z) that if the condition is true (so the condition is satisfied) these links are hidden, otherwise the links are highlighted".

3. CAM

Kees uses the CAM tool to indicate that the Kees_prereq CRT needs to be applied, to concept "sun" (for concept X), and star for concept Y. In the Kees _prereq he has indicated that for concept Y the UM variable knowledge should be used. Hence the course upon deploy will use "star.knowledge".

This means that Kees wants to give access to a page about concept "sun" if the user does have the internal knowledge about the star.knowledge.

3.1.3 Scenario 3 – Nicole and the Solar System application

Student Nicole has to follow the applications Solar System and Space Physics. Her UM data is available in the LMS and in the GUMF.

1. LMS

Nicole accesses to the LMS and she can see the list of courses where she is enrolled in. The list can have some presentation features adapted to her UM data provided by the GUMF and used by the CRT.

Before starting the adaptive course Solar System, the LMS presents a quiz to be filled. The information related to the event "quiz results" is provided to the GUMF (with object 'sun' and predicate 'quizScore').

2. GUMF

The update of the quizScore for Nicole triggers the reasoner plugin, created by Paul. The quizScore for 'sun' is converted to externalKnowledge for 'sun'.

In detail the rule states that for: - Subject: a person/user

- Predicate: quizScore

- Object: sun

- Level: 1-5

a new statement will be generated with:

- Subject: same as above

- Predicate: externalKnowledge

- Object: sun (as above)

- Level: the above level converted to a 0-100 scale.

3. LMS

Nicole selects the adaptive course Solar System and the LMS launches the application from GALE. The information related to the event "access to a course" will be eventually provided to the GUMF.

4. GUMF

The GUMF manages the update of UM information about Nicole.

5. GALE

GALE provides the application updated with Nicole's UM info provided by the GUMF.

6. GVIS

Nicole wants to check her progress with the adaptive course. She can see in the LMS interface a **visual indicator** (widget) that presents a simplified view of her progress with the adaptive course. Moreover, other compact indicators are provided: the overall level of knowledge, compared to the overall level of the class, and the target level defined by Paul.

Based on the information presented by this initial aggregated indicator, Nicole is able to understand her personal situation on the adaptive course and is stimulated to explore her analytic profile. The **analytic profile** presents a set of detailed view of the learner's performance data. In particular, she can see the list of concepts of the adaptive course. For each of these concepts, the visualization reports **the level of knowledge**, the **level of knowledge of the class**, and the **target level**. Using these graphical representations that provide a fine level of details of concepts of the adaptive course, Nicole reflects on her learning status, identifies her strengths and weaknesses, and achieves a deeper understanding of the personal knowledge.

7. LMS

When Nicole exits from the adaptive course, she is back to the list of courses where she is enrolled. The list can have some presentation features adapted to her UM data provided by the GUMF, such as the icon annotation that indicates which adaptive courses have been completed successfully or not.

References

1. Oneto L. et all: D7.1b - Updated specification of the operational infrastructure
2. Simon, B. and all, A Simple Query Interface for Interoperable Learning Repositories *WWW 2005* , May 10-14, 2005.
3. Steiner C., Nussbaumer A.: D3.2a - Integrated model of adaptation on learning with specifications
4. Hendrix M., Cristea A.: D3.3a - Design of a CAM definition tool
5. van der Sluijs K. et al.: D1.1a - CAM to adaptation rule translator – specification
6. Scalable Vector Graphics (SVG) 1.1 Specification, W3C Recommendation 14 January 2003, http://www.w3.org/TR/SVG11/
7. Stash, N., Cristea, A.I., and De Bra, P., Explicit Intelligence in Adaptive Hypermedia: Generic Adaptation Languages for Learning Preferences and Styles, Proceedings of the HT 2005 CIAH Workshop, Salzburg, 2005
8. Cristea, A.I., De Bra, P., Explicit Intelligence in Adaptive Hypermedia: Generic Adaptation Languages for Learning Preferences and Styles, HT 2005 CIAH Workshop, Salzburg. (2005)
9. Oneto L., et all: D3.1 - Design specification of a DM definition tool
10. Hendrix M., et all: D3.3b - Initial implementation of the CAM definition tool
11. Herder, E.: Forward, Back and Home Again - Analyzing User Navigation on the Web. Ph.D. Thesis, University of Twente. ISBN 907383873-8. (2006)
12. Herder E., Abel F.: D2.1 - Definition of an appropriate User profile format
13. Herder E.: D2.3 - Tool for reasoning about user observations
14. van der Sluijs K. at all: D1.1b - CAM to adaptation rule translator – implementation
15. Oneto L., et all: D5.2a - Conversion models between GRAPPLE and LMSs
16. IMS Learner Information Packaging Information Model Specification, January 2005, http://www.imsglobal.org/profiles/
17. D5.2b - Conversion components between GRAPPLE and LMSs.