

GRAPPLE

D5.3c Version: 1.0

Extensions and modifications of learning specifications and LMSs focused on adaptive learning

| | |
|----------------------------|---|
| Document Type | Deliverable |
| Editor(s): | Daniel Burgos |
| Author(s): | Daniel Burgos, Noaa Barak, Vicente Romero (ATOS), Christian Glahn, Marcus Specht, Marion R. Gruber, Dominique Verpoorten, Sebastian Kelle, Roland Klemke (OUNL), Maurice Hendrix, Alexandra Cristea (Warwick) |
| Reviewer(s): | Patrick Pekczynski (IMC), Alexander Nussbaumer and Christina Steiner (UniGraz) |
| Work Package: | WP5 |
| Due Date: | M30 |
| Version: | 1.0 |
| Version Date: | 1-9-2011 |
| Total nr. of pages: | 49 |

Abstract: This deliverable follows the research started in D5.3a and D5.3b. In the previous deliverables the background about IMS Learning Design has been showed and how to model adaptive learning with this specification. In addition, a thorough analysis of a number of use cases has been provided and a detailed list of issues to be modified and improved in the specification to better express adaptation. Based on these outcomes the report at hand provides recommendations, modifications and extensions to IMS Learning Design in order to improve its expressiveness of adaptive learning. Finally, an effective connection between Grapple and IMS-LD has been investigated.

Keyword list: IMS Learning Design, Unit of Learning, Template, recommendations, extensions, modifications

Summary

The work presented in this deliverable concludes the work of task 5.3. The constraints identified previously in 5.3b have been used in order to create a set of recommendations, using extensions and modifications, to promote the ability of the IMS-LD expressiveness for a better support of adaptive learning processes. The recommendations have been categorised in two groups: Modelling and Architecture. Modelling is focused on process, components and programming resources of IMS-LD. Architecture is focused on the communication channels of IMS-LD, in both ways and it deals with upper layers of the specification, beyond modelling issues. Modelling and Architecture issues need to be addressed in order to improve the pedagogical expressiveness and the integration of IMS-LD. Additionally an orchestrated solution has been provided that meets these goals. A structured and organised group of modifications and extensions of IMS-LD has been developed, which matches the various issues reported in D5.3b. Modifications, extensions and addition of different elements have been suggested, aiming at the strength of the specification on adaptive learning processes, along with general interest issues (e.g. interoperability). In addition, a comparison between the LAOS model and the IMS-LD specification was made, in order to look for similarities in high and low level features that allow for an effective connection between Grapple and IMS-LD.

Authors

| Person | Email | Partner code |
|-----------------------------|--------------------------------|--------------|
| Daniel Burgos | daniel.burgos@atosresearch.eu | ATOS |
| Noaa Barak | noaa.barak@atosresearch.eu | ATOS |
| Vicente Romero | vicente.romero@atosresearch.eu | ATOS |
| Christian Glahn | christian.glahn@ou.nl | OUNL |
| Marcus Specht | marcus.specht@ou.nl | OUNL |
| Marion R. Gruber | marion.gruber@ou.nl | OUNL |
| Dominique Verpoorten | dominique.verpoorten@ou.nl | OUNL |
| Sebastian Kelle | sebastian.kelle@ou.nl | OUNL |
| Roland Klemke | roland.klemke@ou.nl | OUNL |
| Maurice Hendrix | maurice@mauricehendrix.co.uk | Warwick |
| Alexandra Cristea | acristea@dcs.warwick.ac.uk | Warwick |

Table of Contents

| | |
|---|-----------|
| SUMMARY | 2 |
| AUTHORS | 2 |
| TABLE OF CONTENTS | 3 |
| TABLES AND FIGURES..... | 4 |
| LIST OF ACRONYMS AND ABBREVIATIONS..... | 4 |
| 1 TASK AND DELIVERABLE DESCRIPTION..... | 6 |
| 2 INTRODUCTION | 6 |
| 3 ANALYSIS OF THE EXPRESSIVENESS OF IMS-LD..... | 7 |
| 3.1 Related work from D5.3b..... | 8 |
| 3.2 A three-focus analysis..... | 10 |
| 3.2.1 Modelling | 10 |
| 3.2.2 Communication, interoperability, integration of Units of Learning..... | 12 |
| 3.2.3 Authoring | 13 |
| 4 RECOMMENDATIONS: EXTENSIONS AND MODIFICATIONS..... | 13 |
| 4.1 General approach to recommendations | 13 |
| 4.2 Design principles..... | 14 |
| 4.3 Modelling..... | 14 |
| 4.3.1 [M.01] Programming structures..... | 17 |
| 4.3.2 [M.02] Synchronization and absolute time | 25 |
| 4.3.3 [M.11] Process model | 28 |
| 4.3.4 [M.12] Enhancing Process Control..... | 29 |
| 4.3.5 [M.13] Prerequisite Conditions..... | 30 |
| 4.3.6 [M.14] State Conditions..... | 32 |
| 4.3.7 [M.15] Activation | 33 |
| 4.3.8 [M.16] History awareness | 35 |
| 4.4 Architecture..... | 37 |
| 4.4.1 [A.01] Better interaction between IMS-LD UoLs and Learning Resources..... | 37 |
| 4.4.2 [A.02] Flexible Service Integration | 41 |
| 5 CONNECTION BETWEEN GRAPPLE AND IMS LEARNING DESIGN | 43 |
| 5.1 Connection to CAM and LAOS | 43 |
| 5.2 LAOS' specifics..... | 44 |

5.3 Comparison of high-level features of LAOS and IMS-LD44

5.3.1 General Content Representation44

5.3.2 Content extent.....45

5.3.3 Generic conceptual point of view45

5.4 Comparison of low-level features of LAOS and IMS-LD.....45

5.4.1 Static content representation45

5.4.2 Adaptation mechanism46

6 CONCLUSIONS AND FURTHER RESEARCH46

7 REFERENCES47

7.1 Papers.....47

7.2 Units of Learning48

DOCUMENT CONTROL ERROR! BOOKMARK NOT DEFINED.

Tables and Figures

List of Figures

Figure 1: Schematic overview of the extended element states29

Figure 2: activity states of the first example32

List of Tables

Table 1: list of state conditions.....33

List of Acronyms and Abbreviations

| | |
|---------|---|
| ADL | Advanced Distributed Learning |
| AEH | Adaptive Educational Hypermedia |
| ALE | Adaptive Learning Environment |
| CAF | Common Adaptation Format |
| CAM | Conceptual Adaptive Model |
| CP | Content Packaging |
| GAL | GRAPPLE Adaptation Language |
| GRAPPLE | Generic Responsive Adaptive Personalised Learning Environment |
| IMS | IMS Global Consortium |
| IMS-LD | IMS Learning Design specification |



D5.3c - Extensions and modifications of learning specifications and LMSs focused on adaptive learning, v1-9-2011

| | |
|-------|---|
| LAG | Layers of Adaptation Granularity |
| LIP | Learner Information Package |
| LMS | Learning Management System |
| QTI | Question and Test Interoperability |
| RTE | Run-Time Environment |
| SCO | Sharable Content Object |
| SCORM | Sharable Content Object Reference Model |
| SN | Sequence and Navigation |
| SS | Simple Sequencing |
| UoL | Unit of Learning |
| VLE | Virtual Learning Environment |

1 Task and Deliverable Description

T 5.3 Modelling with and Extending IMS-LD (ATOS, OUNL, UCL, UCAM, GILABS)

Analysis and modelling of adaptive IMS-LD Units of Learning show adaptive features with identification of gaps between the theoretical approach and the practice. In addition, this task will carry out an implementation of a complete set of use cases and templates showing the different findings and creations. Concrete adaptive learning material will be converted between LMSs so that they can be used in evaluation experiments in WP9 and WP10. Training material for importing, exporting and converting adaptive learning material will be prepared for use in WP11.

The study of IMS-LD may lead to proposals of extensions and modifications of IMS-LD, other specifications and some relevant LMSs to support the best implementation of adaptive features. When deemed appropriate negotiations with standardisation bodies and supporting institutions will be started.

D5.3a Analysis, modelling and implementation of IMS-LD Units of Learning (ATOS, M18)

A full and solid set of Units of Learning developed with IMS-LD will be modelled to show specific features of adaptation, expressed in CAM, and to identify shortcomings and possible ways of improving in the specification. In addition, a rich repository of use cases and templates will be developed (M24), ready to use by end users (authors) dealing with adaptive learning and that need to integrate their adaptive learning material in an LMS.

D5.3b Analysis, modelling and implementation of use cases and templates with IMS-LD (ATOS, M24)

Following the work in D5.3a, a rich repository of use cases and templates will be developed, ready to use by end users (authors) dealing with adaptive learning and that need to integrate their adaptive learning material in an LMS.

D5.3c Extensions and modifications of learning specifications and LMSs focused on adaptive learning (ATOS, M30)

Based on D5.3a and D5.3b input, a set of extensions and modifications will be defined and delivered, very much focused on improving the expressive power of IMS-LD for adaptive learning.

2 Introduction

This report is the result of the final activity within GRAPPLE's WP5 in which the constraints identified in 5.3b have been used. A set of recommendations has been created, using extensions and modifications, to promote the ability of the IMS-LD expressiveness for a better support of adaptive learning processes. Additionally, the research done on adaptation patterns in IMS-LD is reported.

This document begins by summarising and analysing the findings in D5.3b (Section 3) from which to go on to a detailed elaboration on the extensions and modifications for facilitating better support of adaptive learning by IMS-LD. Finally, the work is concluded by offering conclusions and recommendations for future research, which will hopefully serve

in making learning environments more responsive, adaptive and personalised, thus contributing to the core objectives of GRAPPLE.

3 Analysis of the expressiveness of IMS-LD

In “D5.3b Analysis, modelling and implementation of use cases and templates with IMS-LD” ATOS did a thorough research about how IMS-LD models and interprets adaptation. A number of Units of Learning have been designed, modelled, executed and evaluated; five uses cases dealing with adaptive assessment, adaptive authoring, adaptive content and adaptive mentoring, along with an application case implemented in the enterprise ATOS Origin. These UoLs were based on learning scenarios previously designed in D5.3a “Analysis, modelling and implementation of IMS-LD Units of Learning”. All of them are available on the Grapple website (www.grapple-project.org). The list of scenarios and UoLs consists of:

1. Learning Scenario 1. Making a project proposal
2. Learning Scenario 2. A new skill
3. Learning Scenario 3. Human resources live
4. Learning Scenario 4. Learning at the workplace
5. Learning Scenario 5. Getting some expertise
6. UoL. Use case: adaptation on the learner’s performance and knowledge: Adaptive Assessment
7. UoL. Use case, adaptation on the learning designer’s method: Adaptive Authoring
8. UoL. Use case, adaptation on the learner’s decision: Adaptive Content
9. UoL. Use case, adaptation on the teacher’s decision: Adaptive Mentoring
10. UoL. Application case: industry setting (foundation course on IMS-LD)

Out of this research and modelling efforts a number of findings have been derived focused on the limitations that IMS-LD provides. These findings are mainly focused on adaptive learning processes. However, since this topic cannot be isolated from the overall approach of the specifications, some of the limitations and additional recommendations also address other topics, like interoperability.

Additionally, the OUNL has approached modelling personalisation and adaptation in three different UoLs and two learning scenarios:

11. UoL. Modern Architecture: Skyscrapers and Residential Homes (Level A)
12. UoL. Modern Architecture: Skyscrapers and Residential Homes (Level B)
13. UoL. Dangerous Knowledge Tour on Web-usability
14. Learning scenario 6: Educational Game for Basic Life-Support
15. Learning scenario 7: Location based Information Support for Laboratory Visits

The first two UoLs (#11, #12) analyse different modelling approaches for personalising learning experiences using IMD-LD Level A (Gruber & Glahn, 2010a) and IMS-LD Level B (Gruber & Glahn, 2010b). This work has been reported in Gruber, Glahn, Specht, and Koper (2010). The UoLs primarily focus on adaptive sequencing and service orchestration.

The third UoL addresses personal meta-cognitive support (Verpoorten & Glahn, 2010). Within the domain of web-usability, students are supported in identifying when they build “dangerous knowledge”. Dangerous knowledge is incorrectly learned concepts connected to a learner’s high confidence that these concepts are correct. The UoL primarily focuses on content personalisation, service integration and assessment. UoLs #14 and #15 are theoretical and have not been implemented so far, because of the limitations in the specification. Both UoLs were used to identify specific limitations in the orchestration model of IMS-LD. Additionally, the fourth UoL (#14) addresses the use of IMS-LD for modelling educational games. Through instructional game-design learners are guided through different practices of basic life-support. Depending on the learners’ performance in the game, additional support or new challenges are provided. The main focus of this UoL lies on pattern-based modelling, adaptive sequencing and adaptive assessment. The fifth UoL (#15) contains an educational design that connects educational information services with physical spaces (described in Glahn & Specht, submitted). The unit addresses specific needs of service and device orchestration in mobile learning settings. This is specifically related to device adaptation.

3.1 Related work from D5.3b

Next the main findings are summarised. With regards to the specification itself:

1. The definition of properties and the link through several working XML files is too complicated to become useful
2. The relation between layers and actions is not straightforward and it has to be done interlacing files, through global elements and XML
3. The lack of a richer conditional structure makes the editing of the set of rules more complicated on paper than they actually are from a rational point of view
4. Controlled iterations in the activities are not allowed. Additionally, a closed activity cannot be re-initialised and/or go backwards
5. The monitoring service doesn’t cover any kind of user grouping. Therefore, a user (e.g. either a teacher or a learner) cannot follow the performance of several other users at the same time
6. Questions and answers are not personalised for user; they are identical for all users with the same role
7. The communication between teacher and student is minimal and indirect. They can view the values of properties but there is no other communication service between them
8. There is a lack of flexibility in the input point of changing the itineraries. In the Sequence type, the learning activity with the question always appears at the same place. In the Selection type, the question is always presented after 2 completed learning activities. In case the learning designer/teacher wants to shift this input point, they cannot do so
9. There is no possibility to handle absolute time to start the course and/or a specific activity. Only relative time to the precise time when the instance is created out of the UoL, it is possible
10. There is no possibility to make a connection to an already existing database (for instance, to make a query or to import already enrolled students or teachers). The data

type of connection is not supported. Therefore, every enrolment has to be done by hand or a specific tool has to be run for that

11. Additionally, any connection with the external world is impossible. For instance, a real-time effective communication between an LMS and an IMS-LD UoL is not possible so far, so that in fact they cannot benefit from each other's mutual services and resources. No dispatcher or service has been foreseen in the specification allowing such connection (Moreno et al., 2007)

12. When an executable module is developed with other technologies (Macromedia Flash and PHP, for in-stance), it cannot be integrated with IMS-LD in any way, identifying an interoperability problem. Although IMS-LD is not developed with the intention of supporting such interactivity with users, it could allow for a valid integration with external resources using a layer of communication/dispatcher.

13. A file uploaded from the hard disk of a computer is stored in a file-type property inside the internal database of the engine (CopperCore, in this case). There is no possibility to change the default configuration for storing or retrieving resources. There is no facility to manage those uploads either. Although this is an issue concerning tools too, the core documents of IMS-LD do not provide this information and/or service either

14. IMS-LD does not allow saving information into external files or retrieving information from any external source

15. To perform a dynamic user selection in order to create groups is not possible. The teacher can monitor each user and provide him/her with some feedback on a personal basis. A property could be set up dealing with groups, but these groups should be established before the actual start. However, if the teacher wants to dynamically create a group of students depending on their answers, this is not possible so far. To this extent, groups and roles are the same thing

16. IMS-LD does not allow for recording the user's behaviour; in fact, no measures (i.e., Total Time Needed, Time Before First Move) can be restored or retrieved

17. As a consequence, adaptation based on the user's behaviour cannot be developed using the IMS-LD specification. Additionally, the current state of tooling does not support it either

In addition, with regard to the current engines, a few issues will be highlighted that would support a more powerful use of the specification:

18. Changes on-the-fly are not possible. In case the teacher or the learning designer wants to change the questions for example, the answers or the content of the next activity to be carried out. Every single resource has to be packed in design and publishing time before the actual running of the instance

19. In questionnaires and other forms with fields, the teacher/learning designer cannot modify the number of questions or answers once the UoL has started

20. There is no option to run the UoL (the whole UoL or a part, such as Learning Activity) twice within the same instance. Once a Learning Activity is closed, the user can read it again but the associated learning flow cannot be executed. For instance, after the question to change the itinerary is made in the historic-route, there is no way to go back

21. There is no flexibility to change the content. When the teacher/learning designer wants to keep the same method and the same structure, but he/she wants to change

one single HTML page with some content, the UoL has to be validated and published again. In this case, the learner and the teacher would have to be enrolled and the learning process starts from the very beginning

22. Users cannot be dynamically enrolled within the UoL, once it has started, and they have to be managed by an external tool

3.2 A three-focus analysis

This deliverable analyses how to improve the expressiveness of IMS-LD for modelling adaptive learning processes. To this extent, we identify gaps and issues to be modified and-or extended within the specification. These solutions are mainly required to be compatible to IMS-LD version 1.0 UoLs, since the aim is to maintain previous work in future runtime environments. However, when a major modification is suggested, it is highlighted, as a means for improvement, not as a requirement. Therefore, it is not recommended to change the paradigm, the overall concept of the already released tools, as such; additionally, it is pointed out how the specification and its implementation in actual tools can be evolved towards a better expressiveness.

In general, Level A of IMS-LD provides the basic skeleton and a general framework to work with Units of Learning. It makes 80% of the whole structure. Level C and above all Level B, provide the specification with stronger and more versatile resources. These two upper levels are the actual responsible means to model some of the current learning and teaching challenges (i.e., active learning, collaborative learning, adaptive learning, runtime tracking) (Koper & Burgos, 2005; Burgos et al, 2007a, 2007b).

IMS-LD will benefit from a re-structure and modification of several elements focused on modelling and architecture. It will also improve the overall pedagogical expressiveness, along with specific features on adaptation of learning processes and integration with other specifications, LMSs and learning resources. These are two main objectives of the specification: personalised learning and interoperability. At the same time, IMS-LD would increase the implementation and a wider support if one or several high-level visual authoring tools are developed. Nevertheless, this issue is out of the scope of this research and deliverable, and it deals with research groups and companies working on the adoption of IMS-LD.

The conclusions have been depicted within the same two main blocks that were used to carry out the analysis: modelling (with a special focus on adaptation) and integration. As a result of this work a brief note about authoring tools for instructional design with IMS-LD is also provided.

3.2.1 Modelling

With regard to general modelling of educational processes and modelling focused on adaptive learning, IMS-LD offers a framework that allows practitioners to formally express simple educational processes. The basic structure of IMS-LD can be understood by non-technical instructional designers and can be used to create simple interactive instructional designs for technology-enhanced and blended learning. However, in more complex or non-standard educational scenarios (i.e. adaptive content, personalised learning itineraries and other topics in Deliverables D5.3a and D5.3b) the modelling process quickly becomes quite challenging. The reasons for this are manifold, but are summarised by the following key concerns:

- The conceptual model is clear: play, acts, roles, role-parts, and etcetera. But all of them, interlaced in a whole structure of learning, become complex. Even the

simplest scenario requires some knowledge of the specification in a technical way. And this is far from being user-friendly, moreover when the usual target people consists of non-technical profiles (Burgos, 2008)

- The orchestration of educational processes itself is a complex task of managing interwoven aspects, such as learning resources, learning environments, learning activities, roles and learner support. Teachers, instructional designers and other educational practitioners use instructional design techniques to provide rules for the efficient handling of this complexity.
- Creating a formal process model for an instructional design that can be (partially) automated and (potentially) repeated requires the ability to create a coherent process structure and a rule-logic for managing this structure. While educational practitioners often struggle to formalise the rules of their instructional designs, technical-support staff appears to have difficulties with model-based authoring concepts.
- IMS-LD's semantic structure has been constructed around an activity concept that directly supports some modelling tasks, but requires work-around solutions for others. Identifying a modelling problem in a specific situation and selecting an appropriate and well-structured work-around for it requires deep technical knowledge of the specification and related runtime implementations.
- The notation itself follows a usual XML Schema and the definition of the several elements and components of the spec can turn too complex, even for skilled programmers. The description of activities, activity structures, environments, etcetera and the long cascade of relationships amongst them, makes a difficult-to-trace chain out of a simple scenario. Not to mention when several roles are involved, when some components of Level B are used or when adaptive processes are required. The programming structure is quite easy, but the combination of elements, components and metaphor, makes it difficult to implement.
- The programming components provided by IMS-LD are quite simple (i.e., simple condition, based arithmetic, visualisation of variables, visibility, DIV layers, etcetera). On the other side, their syntax is long, which hinders the rationale of the modelling process itself
- Many problems of creating educational designs are related to the choice and current development of the authoring tools. As many of the educational design tools are not intended to be exploited in commercial settings, these tools are often not fully ready for the end-user. This can be found with authoring tools for IMS-LD, which often focus more on exposing specification related aspects of an instructional design than on supporting the modelling process itself. On the other hand, other instructional design tools are bound to the educational model of one runtime environment.

From these issues can be concluded that some of the challenges related to modelling educational designs are directly related to the IMS-LD specification, while others concentrate on the authoring tools and engines. The present IMS-LD specification tackles the complexity of the domain of educational process modelling.

The work on including adaptation and personalisation concepts into the modelling process helped to identify and isolate various settings in which the IMS-LD concept and the IMS-LD semantic structure add additional complexity or external (non-interoperable)

dependencies to the educational design process. This additional complexity cannot be entirely hidden by the logic of authoring tools without losing flexibility or interoperability. This overhead is certainly a barrier that hinders the integration of adaptation and personalisation concepts into the educational practice. It is therefore necessary to identify solutions for simplifying the modelling concepts and for extending the expressiveness of the IMS-LD semantics.

Within the scope of this general objective five working areas that address the modelling of adaptive and personalised learning processes with IMS-LD have been identified:

1. The process model of IMS-LD supports only two states: completed and non-completed. A more complex process model to reflect all stages of a learning activity would simplify the modelling of several approaches to personalisation and adaptation.
2. A consequent process control model across all aspects of an instructional design is missing. The current process model of IMS-LD emphasises learning activities and has limited semantic support for defining instructional guidelines at the level of learning environments and learning roles. This affects a more dynamic approach to domain model triggered processes and to explicit rules for managing the assignment to roles.
3. The current property model is based on a state-based process model. This model does not provide support for learning history. This primarily affects the user modelling and concept adaptation modelling support.
4. The perspective of learning resources in IMS-LD is too closely related to the learning design. More flexible approaches to connect process properties to interactive elements would allow the modelling of more adaptive interactive resources.
5. Background service integration is only partially supported by IMS-LD. Integrating support of social and collaborative software in instructional designs is therefore difficult or impossible to model appropriately.

3.2.2 Communication, interoperability, integration of Units of Learning

Three ways of communication have been studied: 1) simple link between parts, 2) embedded information packages with no information exchange and 3) full communication of information packages, sharing variables and states. This third solution becomes the most effective one. It implies the development of a communication layer that deals with effective bi-directional exchange of data between information packages. This solution also allows for the communication and sharing of services, along with variables, values and states, between IMS-LD and any outside counterpart, i.e., other specifications (SCORM), languages (PHP, Java, Action Script), and LMSs (LAMS, Moodle, .LRN).

Should this exchange actually happen, it will encourage the re-use of information packages in different contexts and the development of templates. It will support and foster the re-purpose of Units of Learning within and amongst the various communities of practice (target groups) that actually use IMS-LD, and not only in those technology-based.

In addition, there is no possibility for a live connection with databases, repositories or any external resource, which allows for information exchange with the UoL.

Although IMS-LD is not responsible, the current two-step working process that makes two isolated parts out of design-time and runtime forces IMS-LD to be compiled and not interpreted. This distinction stops an on-the-fly visualisation and modification of the Extensions and modifications of learning specifications and LMSs focused on adaptive learning

learning design, which would improve the interactive personalisation of the learning process. This issue deals with how IMS-LD is interpreted by tools and engines developers and not with how the specification is actually designed.

3.2.3 Authoring

As mentioned before, this research and deliverables are focused on the specification itself and does not deal with tools. However, authoring environments and engines largely influence what can be modelled and how. That is why two key issues have been pointed out that could support the actual adoption of IMS-LD by the target groups:

- a) There is a need of high-level visual authoring tools. So far, there are two types of authoring environments: effective but too technical, even for technical profiles. Simple to understand but not powerful since they usually deal with the very basic Level A. The educational model of a UoL should be as independent as possible from technical requirements or the underlying elements, components or infrastructure in order to assure the actual modelling and implementation. High-level authoring depends on flexible and yet easy to use visual authoring tools. This implies the integration of specialised tools that target the needs of different stakeholders. A more visual approach will certainly improve the understanding and use of IMS-LD in a broader sense by the target groups. Technical low-level editors should work with the visual high-level editors though, to allow for a fine-grain programming.
- b) Any authoring tool should allow for an integrated modelling, which works with the manifest, the resources and the required external XHTML files with a common interface. It should set properties and inter-relations easily. An integrated perspective on the authoring process will allow for a better modelling process. It should be easy to arrange and re-arrange learning resources, services, roles and activities and to manipulate the rules for orchestrating the different parts of a UoL.

4 Recommendations: Extensions and modifications

4.1 General approach to recommendations

A technical set of solutions has been developed based on the previous analysis and conclusions. These solutions deal with extensions, modifications and addition of modelling structures, elements and components and the architecture of IMS-LD. More specifically:

- a) A set of solutions focused on general-purpose modelling has been developed. These elements will be used as part of others specifically used in learning processes, like personalisation. Furthermore, they become a basic set to be re-purposed in different contexts and goals.
- b) A set of solutions focused on adaptive learning has been developed. A few very specific processes cannot be approached with just general structures. They need on-purpose elements which come across on-purpose goals on personalisation.
- c) A set of solutions that facilitate and improve the integration of Units of Learning and a bi-directional communication with other external resources, systems and standards has been developed. When needed, the need for a way of communication (e.g., a communication layer) will be highlighted although its development is something outside of the scope of this deliverable. The focus is on the specification itself and how to improve the pedagogical expressiveness and not on building any ad hoc technical artefact.

Two key design principles are underlying the suggestions made in the following sections.

- a) All suggestions follow the same concepts that were introduced by IMS-LD version 1.0. This implies that all suggestions for improvement have to interplay safely with the current version of the IMS-LD specification. In specific cases however, a few elements should be modified to simplify the structure and-or to improve the performance. These must be taken as proposals, not requirements.
- b) The enhancements and improvements need to reduce the complexity of the modelling principles whenever possible. This implies that an enhancement should only add new features to the IMS-LD semantics but it should also simplify the modelling process.

This project also concentrated on solutions leaning on two categories: a) **Modelling**, that compiles every single extension, modification or addition, general or specific, to the specification and the information model; and b) **Architecture**, that deals with functional requirements of the spec, with a focus on the interoperability, communication and integration of IMS-LD with other external means. In both cases, the aim was to find the highest performance along with the minimal structural change. Furthermore, the original specification has been respected as much as possible and as few changes as possible were made. On the other side, they all are needed to build the suggested solution and cope with the overall approach.

4.2 Design principles

IMS-LD provides semantics for modelling instructional designs. Instructional designs are rules for controlling learning processes. IMS-LD is therefore considered as a process modelling language and not a programming language. IMS-LD is indeed an XML-based specification and not a language for programming. Some aspects of process modelling are subject to other specifications. A process modelling language should not interfere with other specifications but provide transparent interfaces to them.

Next the two main categories of the recommendations will be described: Modelling and Architecture.

4.3 Modelling

This subsection presents a rich and structured set of recommendations, modifications and extensions to improve the expressiveness of IMS Learning Design on adaptive learning processes. It lays on the aforementioned analysis. The following set of tables show a summary of the constraints, analysis and recommendations. The tables are structured as follows: in the grey-coloured, first row of each table, Column 1 (ID) numbers the constraints and analysis issues. Prefix M relates to issues concerning Modelling and prefix A relates to issues concerning Architecture. Column 2 (Constraints...) provides a description of those issues numbered in Column 1. The white-coloured row(s) after that presents the recommendation/s in the same couple format: ID and description (Burgos, 2010, submitted):

| ID | Constraints, analysis and recommendations |
|-----------|--|
| [M.01] | Programming structures and resources are very basic (simple condition, simple arithmetic, properties set-up, visibility, DIV layers) |
| [Rec.01a] | Condition type case |

| | |
|-----------|---|
| [Rec.01b] | Condition type case with automatic ranges |
| [Rec.01c] | Conditional loop, type while |
| [Rec.01d] | Integer loop, type for-next |
| [Rec.01e] | Modification of the element <calculate> |

| | |
|----------|--|
| [M.02] | There is no management of absolute time. There is no synchronisation nor input point to work with relative time from |
| [Rec.02] | Modification of reference to relative time. Addition of reference to absolute time |

| | |
|----------|---|
| [M.03] | Notification service, in Level C, is under-used. It only sends an email or plays an activity |
| [Rec.03] | Extension of the notification service, beyond using <i>sendmail</i> and playing an activity. It can be called from other structures besides the <on-completion> part of a learning activity |

| | |
|----------|--|
| [M.04] | There is a blurred way to handle the definition and use of properties and links amongst the several XML with global elements |
| [Rec.04] | Syntax modification, definition and use of elements view-property and set-property, as long as the properties which make use of them |

| | |
|----------|---|
| [M.05] | Relationship between DIV layers and the visibility property is difficult to make and follow |
| [Rec.05] | In principle, the visibility property of any layer is turned off (hide), making the conditional structure simpler |

| | |
|----------|--|
| [M.06] | There is no possibility for iterations in any of the basic structures of the IMS-LD metaphor (i.e. learning activity, support activity, activity structure, act, play) |
| [Rec.06] | Extension of the current syntax of every element with a parameter <iteration> which defines a integer loop (type for-next) and-or a conditional loop (type while) |

| | |
|----------|--|
| [M.07] | There is no synchronisation input point in the manifest |
| [Rec.07] | Addition of an element GOTO which allows for a direct guiding of the learning flow |

| | |
|-----------|--|
| [M.08] | There is no possibility to assign a specific activity to a selected user |
| [Rec.08a] | Addition of an element ASSIGN-ACTIVITY-TO-USER which allows for a direct match amongst users, groups and roles, with learning activities and activity structures |

D5.3c - Extensions and modifications of learning specifications and LMSs focused on adaptive learning, v1-9-2011

| | |
|-----------|--|
| [Rec.08b] | Addition of an element ASSIGN-USER-TO-ACTIVITY which allows for a direct match amongst users, groups and roles, with learning activities and activity structures |
| [Rec.08c] | Addition of an element SWITCH-ACTIVITY which allows for turning activities on-off and activity structures |

| | |
|----------|--|
| [M.09] | There is no possibility to make groups out of a selection inside the instance |
| [Rec.09] | Addition of an element CREATE-GROUP which allows for grouping users of the same role |

| | |
|----------|--|
| [M.10] | The monitoring service does not allow for monitoring of groups |
| [Rec.10] | Extension of the monitoring service to trace roles and groups |

| | |
|----------|--|
| [M.11] | The process structure of IMS-LD version 1.0 is based on a state principle. This state principle is linked to learning activities. Each learning activity in a UoL can have two distinct states: completed or not completed. In addition to the two process states every learning activity, learning environment, learning resource or content fragments can be visible or invisible. |
| [Rec.11] | In order to improve the process model and reduce the need to work around the current limitations using show and hide operations, the following four states are proposed to match the different phases of a process lifetime: available, active, visited, completed |

| | |
|----------|---|
| [M.12] | Process Control is not tight enough |
| [Rec.12] | All relevant IMS-LD elements should be included in the process control using unique semantics. This simplifies the modelling of process control structures for orchestrating different aspects of instructional designs |

| | |
|----------|--|
| [M.13] | The “availability” of the related learning-design item is not checked |
| [Rec.13] | It is proposed to add a “prerequisite-condition” element to IMS-LD elements in order to control the “availability” of the related learning-design item. The “prerequisite-condition” is similar to the “complete-activity” element. Additionally, it is proposed to introduce a ‘completed-condition’ also to role, environments, learning objects and services. This would extend the process model beyond the activities |

| | |
|----------|--|
| [M.14] | Actions cannot be triggered based on the state of an activity |
| [Rec.14] | For IMS-LD Level B it is proposed to add a ‘has-state’ condition. This conditional statement allows defining |

| | |
|----------|---|
| | triggers that are based on the state of an activity |
| [M.15] | IMS-LD elements can be activated by user-choice only |
| [Rec.15] | For this purpose it is proposed to introduce an “activate” element for operand expressions. Similar to the has-state condition, the activate- and deactivate-operands have a “scope” attribute that defines the range of the activation and has the same values as for the has-state condition |
| [M.16] | There is no history awareness |
| [Rec.16] | Learning history is a complex problem with regard to IMS-LD. For supporting meta-reflection the learning history plays an important role. Handling the following factors without introducing complexity of set operations to the modelling language is suggested: First attempt timestamp, Last attempt timestamp, and Number of attempts. In addition, it is suggested to offer a unified way to access the historic information, because all elements of a UoL follow the same model |

Next, a few relevant recommendations, with description, syntax and example code-snippets will be detailed when needed.

4.3.1 [M.01] Programming structures

| ID | Constraint |
|--------|--|
| [M.01] | Programming structures and resources are very basic (simple condition, simple arithmetic, properties set-up, visibility, DIV layers) |

| | |
|--|---------------------------------|
| Recommendations for modification and-or extension | |
| [Rec.01a] | Conditional structure type CASE |
| Description | |
| It allows for multiple actions, distributed by conditions which depend on a single property. Unlimited number of actions <case-conditions> represents any operator (i.e. greater-than, less-than, is..) used by the structure case to execute the related actions; an alternative action is included under else. Comparisons can be made against either properties (ID-PROPERTY) or specific values (VALUE) <action> represents the set of (x=1..n) individual actions to execute, type condition, notifications, change-of-property, calculate, and etcetera. These actions are represented by [Ax] | |
| Syntax | |
| <pre><case> <property-ref ref="ID-PROPERTY"/> </case></pre> | |

```

<case-conditions>
  <greater-than> <!-- greater-than, less-than, is -->
    <property-ref ref="ID-PROPERTY"/>
  </greater-than>
  <action>
    [A1..n] <!-- show/hide,change-property,notification...--
>
  </action>

  <less-than>
    <property-value>VALUE</property-value>
  </less-than>
  <action>
    [An+1..m]
  </action>

  <!-- ... -->

  <else>
    <action>
      [Am+1..p]
    </action>
  </else>
</case-conditions>

```

Example (*in italic,related semantics; in **bold**, recommendation*)

```

<!-- ... -->

<loc-property identifier="score">
  <datatype datatype="integer"/>
  <initial-value>0</initial-value>
</loc-property>

<loc-property identifier="threshold">
  <datatype datatype="integer"/>
  <initial-value>5</initial-value>
</loc-property>

<!-- ... -->

```

```

<learning-activity identifier="passed">
  <title>You passed</title>
  <activity-description>
    <item identifierref="res-passed" identifier="I-passed" />
  </activity-description>
</learning-activity>

<learning-activity identifier="not-passed">
  <title>You did not passed</title>
  <activity-description>
    <item identifierref="res-not-passed" identifier="I-not-passed"
/>
  </activity-description>
</learning-activity>

<!-- ... -->

<case>
  <property-ref ref="score"/>
</case>

<case-conditions>
  <less-than>
    <property-ref ref="threshold"/>
  </less-than>
  <action>
    <show>
      <learning-activity-ref ref="not-passed" />
    </show>
  </action>
  <greater-than>
    <property-value>VALUE</property-value>
  </greater-than>
  <action>
    <show>
      <learning-activity-ref ref="passed" />
    </show>
  </action>
  <else>

```

```

<!-- You are right on the threshold, so you get both activities -->

        <action>
        <show>
            <learning-activity-ref ref="not-passed" />
            <learning-activity-ref ref="passed" />
        </show>
        </action>

    </else>
</case-conditions>

```

| Recommendations for modification and-or extension | |
|--|------------------------------|
| [Rec.01c] | Conditional loop, type WHILE |
| Description | |
| <p>Loop keeps iterating until the property meets the condition</p> <p>[condition] represents any of the current conditions in IMS-LD (greater-than, less-than, is...)</p> <p><action> represents a set of x=1..n individual actions to be executed</p> <p>A break command is included to force the termination of the loop</p> | |
| Syntax | |
| <pre> <when> [condition] <action> [A1] [A2] ... <break> ... [An] </action> </when> </pre> | |
| Example (in italic,related semantics; in bold, recommendation) | |
| <pre> <!-- ... --> <locpers-property identifier="end-intro"> <datatype datatype="boolean"/> <initial-value>1</initial-value> </locpers-property> <locpers-property identifier="state"> </pre> | |

```

        <datatype datatype="string" />
        <initial-value>Welcome to the quiz!</initial-value>
</locpers-property>

<!-- ... -->

<learning-activity identifier="quiz">
    <title>How much do you know?</title>
    <activity-description>
        <item identifierref="res-quiz" identifier="I-quiz" />
    </activity-description>
    <complete-activity>
        <user-choice/>
    </complete-activity>
</learning-activity>

<!-- ... -->

<when>
    <is-not>
        <property-ref ref="end-intro"/>
        <property-value>1</property-value>
    </is-not>
    <action>
        <show>
            <learning-activity-ref ref="quiz" />
        </show>
        <change-property-value>
            <property-ref ref="state"/>
            <property-value>Still in!</property-value>
        </change-property-value>
        <notification>
            <!-- ... -->
        </notification>
    </action>
</when>

```

Recommendations for modification and-or extension

[Rec.01e] Modification of the structure <calculate>

| Description |
|---|
| <p>Basic arithmetic operations are allowed, in a sequential, not nested way It allows using the same operator in a sequence. It allows grouping calculations As an extension, it adds average and percentage, to the basic current operators</p> |
| Syntaxis |
| <pre> <!-- operatorX could be sum, subtract, multiply, division --> <!-- operatorX could also be average, percent, but with their specific syntax --> <calculate> <!-- "ID-PROPERTY" operator1 VALUE operator2 "ID-PROPERTY-2" --> <property-ref ref="ID-PROPERTY" /> <operator1> <property-value>VALUE</property-value> </operator1> <operator2> <property-ref ref="ID-PROPERTY-2" /> </operator2> </calculate> <calculate> <!-- "ID-PROPERTY" operator1 VALUE operator1 "ID-PROPERTY-2" --> <operator1> <property-ref ref="ID-PROPERTY" /> <property-value>VALUE</property-value> <property-ref ref="ID-PROPERTY-2" /> </operator1> </calculate> <calculate> <!-- "ID-PROPERTY" operator2 ("ID-PROPERTY-2" operator1 VALUE) --> <property-ref ref="ID-PROPERTY-2" /> <operator2> <group-subtotal ref="SUB-1"> <property-ref ref="ID-PROPERTY" /> <operator1> <property-value>VALUE</property-value> </operator1> </group-subtotal> </operator2> </pre> |

```

</calculate>

<calculate>
<!-- AVERAGE of ("ID-PROPERTY" + VALUE + "ID-PROPERTY-2") -->
  <average>
    <property-ref ref="ID-PROPERTY" />
    <property-value>VALUE</property-value>
    <property-ref ref="ID-PROPERTY-2" />
  </average>
</calculate>

<calculate>
<!-- "ID-PERCENTAGE" of "ID-BASE" -->
  <percent>
    <property-ref ref="ID-BASE" />
    <property-ref ref="ID-PERCENTAGE" />
  </percent>
</calculate>

```

Example (in *italics*, related semantics; in **bold, recommendation)**

```

<!-- ... -->

<locpers-property identifier="ID-OP-1">
  <datatype datatype="integer"/>
  <initial-value>7</initial-value>
</locpers-property>

<locpers-property identifier="ID-OP-2">
  <datatype datatype="integer"/>
  <initial-value>5</initial-value>
</locpers-property>

<locpers-property identifier="ID-OP-3">
  <datatype datatype="real"/>
  <initial-value>54.0</initial-value>
</locpers-property>

<locpers-property identifier="ID-PERCENTAGE">
  <datatype datatype="real"/>
  <initial-value>20.0</initial-value>
</locpers-property>

```

```

<!-- ... -->

<calculate>
<!-- "ID-OP-1" + 3 - "ID-OP-2" -->
  <property-ref ref="ID-OP-1" />
  <sum>
    <property-value>3</property-value>
  </sum>
  <subtract>
    <property-ref ref="ID-OP-2" />
  </subtract>
</calculate>

<calculate>
<!-- "ID-OP-1" + 3 + "ID-OP-2" -->
  <sum>
    <property-ref ref="ID-OP-1" />
    <property-value>3</property-value>
    <property-ref ref="ID-OP-2" />
  </sum>
</calculate>

<calculate>
<!-- "ID-OP-1" * ("ID-OP-2" + 3) -->
  <property-ref ref="ID-OP-1" />
  <multiply>
    <group-subtotal ref="SUB-1">
      <property-ref ref="ID-OP-2" />
      <sum>
        <property-value>3</property-value>
      </sum>
    </group-subtotal>
  </multiply>
</calculate>

<calculate>
<!-- ("ID-OP-1" + 3 + "ID-OP-2") / 3 -->
  <average>
    <property-ref ref="ID-OP-1" />

```

```

        <property-value>3</property-value>
        <property-ref ref="ID-OP-2" />
    </average>
</calculate>

<calculate>
<!-- "ID-PERCENTAGE" * "ID-OP-3" / 100 -->
    <percent>
        <property-ref ref="ID-OP-3" />
        <property-ref ref="ID-PERCENTAGE" />
    </percent>
</calculate>

```

4.3.2 [M.02] Synchronisation and absolute time

| ID | Constraint |
|--------|--|
| [M.02] | Relative execution time is not linked to any element of synchronisation (e.g. activity or act). There is no reference to absolute time |

| Recommendations for modification and-or extension | |
|---|---|
| [Ext.04] | Reference to relative time is modified, so it can be assigned to specific elements. Reference to absolute time is added |
| Description | |
| <p>An act, a play, a learning activity, a support activity and a notification, can be linked to a relative timestamp, so that synchronisation between different elements and user actions can be orchestrated. This link is compatible with the current timestamp related to the actual start of the instance</p> <p>A reference to absolute time is added, as a new resource for synchronisation</p> | |
| Syntax | |
| <pre> <!-- Every record <time-limit> in a UoL, a play, an act or an activity can be link to a specific reference to a relative time --> <complete-unit-of-learning> <!-- in 2 months --> <time-limit>P2M</time-limit> </complete-unit-of-learning> <complete-play> <!-- in 1 month and 25 days --> <time-limit>P1M25D</time-limit> </pre> | |

```

</complete-play>

<complete-act>
<!-- in 1 month and 20 days -->
    <time-limit>P1M20D</time-limit>
</complete-act>

<complete-activity>
<!-- in 1 month and 15 days -->
    <time-limit>P1M15D</time-limit>
</complete-activity>

<!-- In addition, the reference to the time-limit can be based on
another item's completion. For instance, about a learning activity: -->

<complete-activity>
    <time-limit>P1M15D</time-limit>
        <!-- 1 month, 15 days, from end of "ID-LA-previous" -->
    <learning-activity-ref ref="ID-LA-previous"/ >
</complete-activity>

<!-- In the same way, a reference to absolute time is included -->

<time-limit-abs>P2007Y2M17D</time-limit-abs> <!-- February 17th,2007 -->

```

Example (in italic,related semantics; in bold, recommendation)

```

<!-- ... -->

<locpers-property identifier="ID-PROPERTY-TIMESTAMP">
    <title>Timestamp</title>
    <datatype datatype="string"/>
    <initial-value>P2007Y2M17D9H</initial-value>
        <!-- February 17th, 2007, 9AM -->
</locpers-property>

<!-- ... -->

<learning-activity identifier="ID-LA-previous" isvisible="true">
    <title>AboutRelativeTime</title>

```

```

    <activity-description>
      <title>CheckingTime</title>
      <item identifier="item-LA-previous"
        identifierref="res-LA-previous" isvisible="true" />
    </activity-description>
  </complete-activity>
</learning-activity>

<learning-activity identifier="ID-LA-1" isvisible="true">
  <title>AboutRelativeTime</title>
  <activity-description>
    <title>CheckingTime</title>
    <item identifier="item-LA-1"
      identifierref="res-LA-1" isvisible="true" />
  </activity-description>
  <complete-activity>
    <time-limit>P1M15D</time-limit>
    <!-- 1 month, 15 days, from end of "ID-LA-previous"
-->
    <learning-activity-ref ref="ID-LA-previous"/ >
  </complete-activity>
</learning-activity>

<!-- ... -->

<act identifier="act-1">
  <title>Act</title>
  <role-part identifier="rolepart-1">
    <title>Role Part</title>
    <role-ref ref="role-student" />
    <learning-activity-ref ref="ID-LA-previous" />
  </role-part>
  <complete-act>
    <time-limit>P10D</time-limit>
    <!-- 10 days from "ID-PROPERTY-TIMESTAMP" --
>
    <property-ref ref="ID-PROPERTY-TIMESTAMP"/ >
  </complete-act>
</act>

```

4.3.3 [M.11] Process model

The process structure of IMS-LD version 1.0 is based on a state principle. This state principle is linked to learning activities. Each learning activity in a UoL can have two distinct states: completed or not completed. In addition to the two process states every learning activity, learning environment, learning resource or content fragments can be visible or invisible.

At IMS-LD Level A the completion state can be altered through condition rules that can be associated to learning activities. If an activity is completed, the objective of that activity has been met and the activity cannot be repeated to improve the result of the task. At IMS-LD Level there are two options for completing a learning activity: user choice and time limit. Alternatively, an instructional designer may decide that an activity cannot be completed in the strict sense of IMS-LD.

In order to improve the process model and reduce the need to work around the current limitations using show and hide operations, the following four states are proposed to match the different phases of a process lifetime.

1. available
2. active
3. visited
4. completed

All states can be represented as Boolean values. This would sustain the current specification of the completed state. In addition to the current states all states should be available to all controllable aspects of a learning design. This means that environments, resources, activity structures, and roles would have process states as well.

The available state means that an instructional design element is available to the learner (or learning facilitator). Within a runtime environment a user has access to those elements that are available to the user.

The active state means that an instructional design element is currently in use by the user. The active state implies the available state is valid as well. Furthermore, if an element is active it is also visited.

The visited state means that an instructional design element has been activated at least once. The visited state is similar to the completed state with the only difference that it locks the result of an activity.

The completed state means that a user has completed an instructional design element. If an element is completed a user can revisit the element but will not be able to alter any previous results. E.g. interactive resources become "read only" resources once they are completed.

Additionally, activation may happen through implicit action. Implicit actions are actions that are possible through resources and services that are not explicitly modelled in the learning design. The following example illustrates the effect of an implicit action.

A learner follows a hyperlink and reaches a resource that is connected to a currently unavailable and inactive environment, the learner gains access to the resource; this will set the available, active, and visited state automatically to valid.

Figure 1 provides a schematic overview of the different proposed lifecycle states of learning design items.

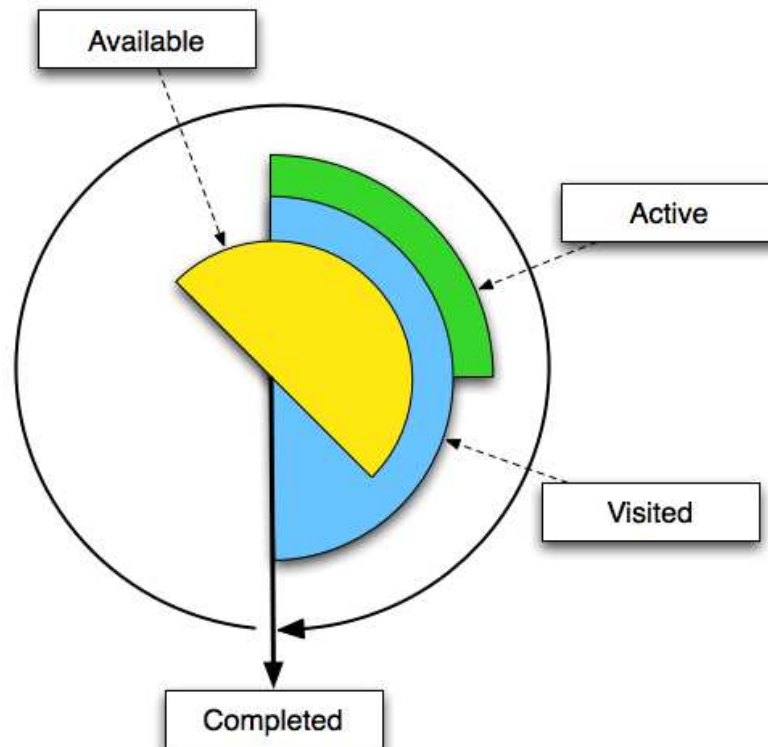


Figure 1: Schematic overview of the extended element states

An explicit available state would eliminate the need of interface level show and hide workarounds for process modelling. Furthermore, the available state fits with the current definition of activity structures.

The visited state is required for fine-grained process control where the “completion” is not required or possible.

The active state is required to identify the currently used design elements for process control. This state comes in handy for modelling synchronous learning activities or tool usage.

In order to benefit from these states it is required to provide conditional semantics for the new process states. The typical location for such semantics would be the conditional structure of IMS-LD Level B.

This process model is underlying the enhancements that are discussed in the following sections.

4.3.4 [M.12] Enhancing Process Control

All relevant IMS-LD elements should be included in the process control using unique semantics. This simplifies the modelling of process control structures for orchestrating different aspects of instructional designs. The following elements are relevant for process control.

- Activity
- Activity-structure
- Role
- Environment

- Learning-object (item)
- Service
- Play
- Act

The IMS-LD specification allows defining the rules for completing activities or activity-structures. The related information model element is “complete-activity”. For IMS-LD Level A these conditions can be “user-choice” or “time-limit”. Level B adds logical conditions to defined detailed completion rules for a learning activity or activity-structure. Through this semantic it is possible to define the end of the lifecycle of a learning activity. In addition IMS-LD provides a semantic element for defining the prerequisites of an activity. This element is the “prerequisites” element. The prerequisites define the availability state of an activity from the perspective of the process model. However, different to the “complete-activity” element, IMS-LD does not allow the definition of conditions for the prerequisites. Instead, the prerequisites are expected as textual information.

Without extending the process control to roles and environments, device adaptation cannot get modelled. This is particularly the case for mobile learning solution with location or context triggers. The main barrier for implementing the UoL “Location based Information Support for Laboratory Visits” is that environments or locations cannot trigger a status change in terms of the modelling language. The current version of IMS-LD allows environments and learning resources only to be visible or hidden. A non-interoperable workaround would be a global personal property that is connected to a condition that triggers a learning activity or makes learning resources available. This global personal property can get set through an external location and presence service. However, the last part of the workaround is dependent on a specific runtime environment and it cannot get represented in the educational design.

4.3.5 [M.13] Prerequisite Conditions

It is proposed to add a “prerequisite-condition” element to IMS-LD elements in order to control the “availability” of the related learning-design item. The “prerequisite-condition” is similar to the “complete-activity” element. Additionally, it is proposed to introduce a ‘completed-condition’ also to role, environments, learning objects and services. This would extend the process model beyond the activities.

During the modelling of three UoLs (Gruber & Glahn, 2010a; 2010b; Verpoorten & Glahn, 2010) it became evident that many process related conditions had to be created because the current version of IMS-LD allows only the modelling of backward process control. This means that it is only possible to define conditions for completing learning activities directly on the activity itself. This makes loops in which learners may return to learning activities as part of a simple educational model. Formal prerequisite conditions would allow to model forward process control that defines the availability of a learning activity. This would provide greater flexibility on modelling educational scenarios without the requirement of ‘completing’ learning activities. In the current version of IMS-LD forward process control is only possible through helper properties that require additional conditions for tracking and process control. This adds a large amount of complexity to other wise simple educational designs. Examples of such additional complexity can be found in (Gruber & Glahn, 2010b).

A prerequisite-condition and completed-condition holds a logical condition that triggers the availability or the completed state of a learning design element. Both elements should be included at IMS-LD Level A and receive advanced semantics from IMS-LD Level B onwards.

The prerequisite-condition elements would allow instructional designers to define formal requirements for a learning activity rather than informal requirements that can be entered in the prerequisite element.

If an element has no prerequisite condition it is considered as always available.

The base semantics of the conditional statements is that all items that are found in the conditional statement have to be valid in order to trigger the condition. This implies that these conditions statements are defined as AND conditions if more than one logical element is found in the condition.

In order to have extended process control it is proposed to include reference elements for conditional validation. For example an author may define an activity-ref, role-ref, or environment-ref inside a prerequisite or completion condition. This would enable instructional designers to link the availability of different aspects of a learning design to other elements. On IMS-LD Level A the additional conditional elements accept only reference elements. These elements are positively validated if the related element has the status completed. This is in line with the current definition of sequencing and selecting processes with activity-structure elements.

The following fragment illustrates the logic for simple activity sequences as a variation of IMS-LD Level A.

```

<learning-activity id="activity_1">
  <title>Starting Activity</title>
  <complete-activity><user-choice/></complete-activity>
  ...
</learning-activity>
<learning-activity id="activity_2">
  <title>Second choice</title>
  <complete-activity><user-choice/></complete-activity>
  <prerequisite-condition>
    <learning-activity-ref ref="activity_1"/>
  </prerequisite-condition>
  ...
</learning-activity>
<learning-activity id="activity_3a">
  <title>First option</title>
  <complete-activity><user-choice/></complete-activity>
  <prerequisite-condition>
    <learning-activity-ref ref="activity_2"/>
  </prerequisite-condition>
  ...
</learning-activity>
<learning-activity id="activity_3b">
  <title>Second option</title>
  <complete-activity><user-choice/></complete-activity>
  <prerequisite-condition>
    <learning-activity-ref ref="activity_2"/>
  </prerequisite-condition>
  ...
</activity>
<activity id="activity_4">
  <title>Over-achiever activity</title>
  <complete-activity><user-choice/></complete-activity>
  <prerequisite-condition>
    <learning-activity-ref ref="activity_3a"/>
    <learning-activity-ref ref="activity_3b"/>
  </prerequisite-condition>
  ...
</activity>
<activity id="activity_5">
  <title>Normal progress option A</title>
  <complete-activity><user-choice/></complete-activity>
  <prerequisite-condition>
    <learning-activity-ref ref="activity_3a"/>
  </prerequisite-condition>
  ...
</activity>

```

```
</prerequisite-condition>
</activity>
```

The logic of this fragment is that instead of using activity structures, the instructional designer models the flow of the learning process entirely based on the prerequisites of the learning activities. It has a sequence that leads to a point where the learner can choose between two learning activities. For over-achieving learners (who do both optional learning activities) this model offers an additional activity that has both optional activities as prerequisite. Learners who only did the first option will be offered an additional learning activity, while learners who did choose the second option are ready (as no more related activities are found). Figure 2 shows the possible activity flows.

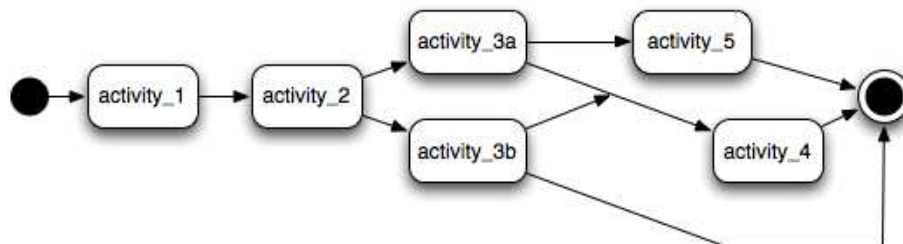


Figure 2: activity states of the first example

As no other information on the state of the prerequisites can be provided at IMS-LD Level A semantics, this fragment requires the prerequisite activities to be marked as completed in order to trigger the prerequisite condition.

4.3.6 [M.14] State Conditions

For IMS-LD Level B it is proposed to add a 'has-state' condition. This conditional statement allows defining triggers that are based on the state of an activity.

The has-state condition can have a 'state' attribute that refers to the process state. The value of the state-attribute can be either "available", "active", "visited", "finished" or "completed". If the value attribute is omitted or contains a different value, the included elements are tested if they are "completed". This is included for backward compatibility with the *complete* condition of the IMS-LD version 1.0 process model. The "finished" state is a pseudo-state that allows instructional designers an easy way of expressing that an element is visited but not active.

The state condition also has an optional scope. By default the scope of the state is tested for the active person only. This default behaviour is the same as setting the scope to "self". The possible values of the scope of the state conditions can be "self" to refer to the active person, "supported-person" to refer to learners that are sharing the play(s) of a facilitator, "role" to refer if the state is set for at least one member of the same role instance of the active person and "all" to refer if the state is set at least once in the current run.

Finally, the has-state condition can hold a reference to the IMS-LD element that has to be tested.

The has-state condition can contain references to learning or support activities, to roles, to learning environments, and to specific items.

Table 1: list of state conditions

| No. | Title | Mult. | Values |
|--------|------------------------|-------|--|
| 1 | Has-state | * | Complex Type |
| 1.1 | state | 0 1 | available, active, visited, finished, completed* |
| 1.2 | scope | 0 1 | self*, supported-person, role, all |
| 1.3 | Learning-activity-ref | * | |
| 1.3.1 | Ref | 1 | IDREF |
| 1.4 | Environment-ref | * | |
| 1.4.1 | ref | 1 | IDREF |
| 1.5 | Role-ref | * | |
| 1.5.1 | ref | 1 | IDREF |
| 1.6 | Item-ref | * | |
| 1.6.1 | ref | 1 | IDREF |
| 1.7 | Support-activity-ref | * | |
| 1.7.1 | ref | 1 | IDREF |
| 1.8 | Activity-structure-ref | * | |
| 1.8.1 | ref | 1 | IDREF |
| 1.9 | Play-ref | * | |
| 1.9.1 | ref | 1 | IDREF |
| 1.10 | Act-ref | * | |
| 1.10.1 | ref | 1 | IDREF |

State conditions would allow the testing of the states of learning activities or learning environments in the UoL (Gruber & Glahn, 2010b). This is a prerequisite requirement for activation of parallel activities or services, which is discussed in the following section.

4.3.7 [M.15] Activation

The default way of activating an LD element is by user choice.

For hierarchical elements, the activation is inherited from their child elements. This means for an activity-structure that it is considered to be active if at least one subordinate activity is active. For environments the active state is inherited from the subordinate items. This means that if one item within an environment is used and is therefore active, the environment is also considered as active.

For roles the active state is determined by the state of the activities of the role-parts and the session status of the related person(s). This means that a role is considered active if a person assigned to the role is currently signed into the run-time environment and has activated an activity that is assigned to the role through a role-part.

Any LD element is automatically deactivated if its prerequisites are no longer met.

Alternatively, an instructional designer has to be able to deliberately activate specific LD elements. To this purpose it is proposed to introduce an “activate” element for operand expressions. Similar to the has-state condition, the activate- and deactivate-operands have a “scope” attribute that defines the range of the activation and has the same values as for the has-state condition.

These operands are useful to model truly synchronous processes and information provisioning. If a hierarchical element is activated (or deactivated), this implies that all subordinate elements are also activated (or deactivated, respectively). This implies that a runtime has to display “synchronous” information next to each other, such as different learner notes in Gruber & Glahn (2010b). This feature is also needed for automated activation, which is needed in mobile and ubiquitous learning settings and augmented reality scenarios.

These operands are subordinate to the prerequisite of LD elements. This means that only those elements will be affected by these operands if they are available to the learner. The following example illustrates the problem.

```

<properties>
  <locpers-property identifier="myproperty">...</locpers-property>
  ...
</properties>
<activities>
  <learning-activity id="activity_1">
    <title>Second choice</title>

    ...

    <environment-ref ref="environment_1"/>
    <environment-ref ref="environment_2"/>
  </learning-activity>
  ...
</activities>
<environments>
  <environment identifier="environment_1">
    <item identifier="learning resource 1"> ... </item>
  </environment>
  <environment identifier="environment_2">
    <item identifier="learning tool 1"> ... </item>
    <item identifier="learning tool 2">
      <prerequisite-condition>
        <is-not>
          <no-value>
            <property-ref ref="myproperty"/>
          </no-value>
        </is-not>
      </prerequisite-condition>
    </item>
  </environment>
  ...
</environments>
...
<conditions>
  <if>
    <has-state state="active">
      <learning-activity-ref ref="activity_1"/>
    </has-state>
    <has-state state="active">
      <environment-ref ref="environment_1"/>
    </has-state>
  </if>
  <then>
    <activate>
      <environment-ref ref="environment_2"/>
    </activate>
  </then>
  <if>
    <or>
      <is-not>
        <has-state state="active">
          <learning-activity-ref ref="activity_1"/>
        </has-state>
      </is-not>
      <is-not>
        <has-state state="active">
          <environment-ref ref="environment_1"/>
        </has-state>
      </is-not>
    </or>
  </if>
</conditions>

```

```

        </or>
    </if>
    <then>
        <deactivate>
            <environment-ref ref="environment_2"/>
        </deactivate>
    </then>
</conditions>

```

The learning activity in this fragment refers to two environments. The first environment (environment_1) contains a learning resource that is required to master the activity. The second environment (environment_2) contains two items that refer to tools that help the learner to handle the learning resource of the first environment. One of the items (learning tool 2) in environment_2 has the prerequisite that the property “myproperty” must be set. Finally, the fragment contains two conditions that define that while environment_1 is active when a learner works on activity_1, environment_2 has to be activated as well.

The effect during runtime will be as described in the following scenario.

Two learners are accessing “learning resource 1” within learning activity “activity_1”. Consequently, both activity_1 and “learning resource 1” are active. Because “learning resource 1” is subordinate to the environment “environment_1”, also environment_1 will change into the active state. Because of this change the condition will validate and trigger the activate operand on environment “environment_2”. This will cause the two subordinate items to be activated as well.

Given that the first learner has the property “myproperty” set while the second learner has this property unset, the two will have access to slightly different environments. For the first learner the prerequisite for the item “learning tool 2” is valid and the item will be activated for the learner, because it is already available. For the second learner this prerequisite is not valid. Consequently, the tool is not activated because it is not available to the learner.

As soon the learners stop working either on the learning resource or if they have completed the learning activity, one or both elements will leave the “active” state. If this happens for the learning resource, also environment_1 will leave the “active” state as no subordinate items are no-longer active. The change of the active state will cause the second condition to validate and to deactivate environment_2.

The main benefit of this extended process model is that it abstracts from the user interface and allows instructional designers to define complex interrelations of learning resources, learning services, learning activities and roles. At the same time this model eliminates the need to reflect interface related aspect in the process definition.

4.3.8 [M.16] History awareness

History awareness has been partially addressed by the UoL “Dangerous Knowledge Tour on Web-usability” (Verpoorten & Glahn, 2010). For supporting meta-reflection the learning history plays an important role. The lack of learning history support in the given UoL leads to the creation of a set of additional local personal properties, which required additional effort to be managed. Mimicking history awareness has been found as highly error prone, because small errors on the additional management of history information easily caused a different runtime-behaviour of the UoL.

Given to the general lack of set operations in IMS-LD, generic history awareness can get reduced to the following factors without introducing complexity of set operations to the modelling language.

1. First attempt timestamp
2. Last attempt timestamp
3. Number of attempts

For properties also the value of the first attempt can be meaningful. The value of the last attempt is always the normal value of a property.

In some cases it is necessary to compare the result of an operation to the previously set data. Currently this is only possible if a temporary property is defined. The following two interfaces would provide a more flexible interface for this task.

4. Previous value
5. Previous value timestamp

Specific to numeric property types, four additional factors can be identified.

6. Maximum value
7. Maximum value timestamp
8. Minimum value
9. Minimum value timestamp

If it is assured that the ordering of restrictions in non-numeric property types reflects an ordinal structure, these properties can be treated as numeric value as well. However, this cannot get assured in all cases.

The first three factors can be applied to all IMS-LD elements including properties. For all IMS-LD process elements (not properties) the first attempt refer to the first time when the element has changed into the active state. Consequently, the last attempt refers to the last time the element has changed into the active state. For these elements the number of attempts refers to the number of times the element changed into the active state.

The value of process elements can also refer to the *duration* from changing into the active state and changing from the active state. As this value is a numeric data type, also maximum and minimum values (and timestamps) can be made available.

It is suggested to offer a unified way to access the historic information, because all elements of a UoL follow the same model. For this purpose such an operand provides a data-value. For consistency reasons it might be useful to have an additional *last-value* operand that would be similar to the property-ref operand but would also work for process elements. This results in 11 operands for accessing history related aspects of design elements.

1. *first-time*
2. *first-value*
3. *last-time*
4. *last-value*
5. *previous-value*
6. *previous-time*
7. *count-times*
8. *maximum-value*
9. *maximum-value-time*
10. *minimum-value*
11. *minimum-value-time*

All operands support the same scope attribute as the activate/deactivate operands. All operands expect exactly one reference to an element of the UoL. For those elements and properties that cannot support specific history information, the value is undefined (is-not).

For properties the first attempt refers to the first time the property is set. The last attempt refers to the last time of the property has been set. The number of attempts refers to the number of times the property has been set. The first value of a property is stored into the first attempts value and is never deleted.

For numeric values the maximum (minimum) value is initialised with the first value and only changed if the new property value is greater (less) or equal then the previous maximum (minimum) value. The timestamp for the respective history variable is set when the value is changed.

For complex arithmetic operations on historic values of properties, it is suggested to outsource these operations to external services. For explicitly integrating such services, semantics for flexible service integration can be applied as it is discussed in the following section.

4.4 Architecture

| ID | Constraints, analysis and recommendations |
|----------|--|
| [A.01] | Interaction between UoLs and learning resources is insufficient |
| [Rec.11] | Provision of an interface layer that allows the exchange of data values and data structures between the learning design and the learning resources |
| [A.02] | Level C provides a only way of communication |
| [Rec.12] | Provision of a flexible service integration in order to improve in-out communication with external resources and services |

4.4.1 [A.01] Better interaction between IMS-LD UoLs and Learning Resources

IMS-LD Level B allows creating interactive UoLs by using the so-called “global elements” inside of learning resources. These elements are directives for a runtime environment to inject special user interface elements that allow a participant in a UoL to change the underlying properties. Alternatively, the lexical naming of properties in resources and in a learning design should be used to exchange the underlying data. This has been previously discussed with regard to the IMS-LD and IMS QTIv2 integration (IMS Global Learning Consortium, 2005).

Both approaches have the obvious drawback that either way an interactive resource is tightly coupled to one learning design. The first approach makes assumptions about the user interface presentation of the data stored in a property. These assumptions are specific to one IMS-LD runtime environment. Neither to the resource designer nor to an instructional designer it is allowed to define guidelines for the presentation of the provided information. For appropriate presentation of the information, the resource designer has to reflect the specifics of the underlying IMS-LD runtime environment. This limits the interoperability of a learning resource across different IMS-LD runtime environments. The global properties also limit the interoperability of learning resources because they rely on

the internal identifiers of the properties. However, most of the recent IMS-LD authoring environments hide this information from the instructional designer. This means that one interactive learning resource is difficult to be used across different UoLs.

The second approach relies on the lexical identity of property identifiers or variable names. Besides that this approach has the same drawback regarding the property identifiers as the global-element handling of the first approach, it also limits the interoperability of the resources in a similar way as the global-elements.

Both approaches have the drawback that a practitioner primarily focuses on content that remains mostly static throughout a run of a UoL. Furthermore, in both cases a fundamental knowledge about the related interfaces is required during the modelling process. The concept of showing and hiding of CSS class names even assumes that all content and services in a UoL share a common style definition that is fully understood by the educational designer.

In the learning scenario about “Modern Architecture” and the “Dangerous Knowledge Tour on Web-usability” the tight coupling between learning resources and the educational design became an imminent problem. This tight interrelation made it necessary to create several resources that all looked similar, but had references to different properties. At the same time the author had limited or no control over the layout and the design of the resources. The continuous switching between resource authoring related problems and educational design related problems caused major confusion, because the two types of authoring are very different activities. Particularly for the Level B UoL about “Modern Architecture” it became necessary to re-author existing resources from a knowledge object repository before they could be used as interactive objects with the UoL.

A similar problem occurred for pattern based modelling techniques while conceptualising the educational game design of the learning scenario “Educational Game for Basic Life-Support”. A pattern can be fully described as a UoL template. Such a UoL is instantiated by assigning domain related content, roles, services, or other pattern instance to the pattern template. Such an authoring approach can simplify the educational design task to arranging instances of educational design patterns. The current version of IMS-LD has basic support for pattern based modelling because it is possible to use existing UoLs instead of learning activities when arranging the educational method. However, given the missing parameter passing, the only way to exchange information between UoLs is through global properties. This workaround makes pattern contextualisation highly difficult to model in IMS-LD.

In order to loosen the relation between resources and the learning design, it is proposed to provide an interface layer that allows the exchange of data values and data-structures between the learning design and the learning resources. This layer should allow instructional designers to define how the information is exchanged.

There is an extensive research on this issue. For instance, in Burgos et al (2006), the authors describe the use of a communication dispatcher to engage a UoL with external games, in order to re-purpose the original use. In addition, in Tattersall et al (2006), the authors deal with the integration between SCORM (ADL, 2000) and IMS-LD and provide the same solution, through an interface layer. Last, in Moreno et al (2007), the authors describe and make the actual implementation of the interconnection between IMS-LD properties and variables in an external application, through a communication layer. This communication dispatcher allows for a live, bi-directional connection with an external resource, in real time. In all these scenarios, the authors suggest the same solution. A runnable UoL shows this approach.

In our current research, three cases can be identified with this regard.

1. Input interfaces that would take information from learning design's properties and expose this information to the learning resource
2. Output interfaces that translate input information of the learning resource into learning design properties
3. Exchange (input-output) interfaces that combine input and output functions.

An input interface passes the value of a property to the interface. The output interface allows the setting of property values through the internal parameters of a resource. This interface prohibits that a resource gets access to the current value of a property. Exchange interfaces are a shortcut to separate input and output interfaces. These interfaces should be used if a property should be altered through the interface and the resource requires information about the property.

All interfaces have to be scoped in order to identify what value has to get passed. The default scope is "self".

The proposed extension of IMS-LD Level B would allow interfaces at the level of learning design items. The following example illustrates this relation.

```
<item identifier="my_resource_item" identifierref="my_resource_object">
  <title>My Learning Resource</title>
  <input parameter="form_parameter" scope="self">
    <property-ref ref="my_property"/>
  </input>
</item>
```

This definition would replace the global element "view-property". The main benefit of this approach is that the interactive components can be designed completely independent from the underlying instructional design.

Some data formats have clear interface definitions for their parameters. These formats are for example IMS QTI tests, Web-forms in the XForm format, Flash animations, and Java applets.

In order to exchange data of a IMS-LD UoL using XForm resources the instructional designer would pass the property information to the named form variables. This is shown as follows.

```
<item identifier="resource_item" identifierref="xform_object">
  <title>Change your data</title>
  <input-output parameter="xformparameter1">
    <property-ref ref="property1"/>
  </input-output>
  <input-output parameter="xformparameter2">
    <property-ref ref="property2"/>
  </input-output>
  <input-output parameter="xformparameter3">
    <property-ref ref="property3"/>
  </input-output>
  <input-output parameter="xformparameter4">
    <property-ref ref="property4"/>
  </input-output>
</item>
```

This would cause the XForm to display the values of property1 – property4 in the respective form fields and allows the participants to change these values. This has several benefits. First, the IMS-LD runtime environment does not require any assumptions about the user interface. This allows resource developers to optimise the interface of interactive resources. Secondly, the same form can be reused for different purposes even within the

same UoL. The same form can also be reused in different UoLs without being changed to the specific local definitions.

The following example shows the reuse of one resource in different environments. This example uses the same XForm objects for knowledge object items in different environments. Normally, this would mean that exactly the same form is presented to the participant. By using the data exchange interface it becomes possible to pass different properties to the form depending on the environment in which the form is used. In the example the first environment the properties “property1” and “property2” are passed to the form, while in the second environment the properties “property3” and “property4” are passed to the form.

```
<environment identifier="environment1">
  <title>First Learning Environment</title>
  <learning-object identifier="LO1" type="knowledge-object">
    <title>First Knowledge Object</title>
    <item identifier="item1" identifierref="xformobject">
      <title>Your Data For Task 1</title>
      <input-output parameter="xformparameter1">
        <property-ref ref="property1"/>
      </input-output>
      <input-output parameter="xformparameter2">
        <property-ref ref="property2"/>
      </input-output>
    </item>
  </learning-object>
</environment>
<environment identifier="environment2">
  <title>Second Learning Environment</title>
  <learning-object identifier="LO2" type="knowledge-object">
    <title>Second Knowledge Object</title>
    <item identifier="item2" identifierref="xformobject">
      <title>Your Data For Task 2</title>
      <input-output parameter="xformparameter1">
        <property-ref ref="property3"/>
      </input-output>
      <input-output parameter="xformparameter2">
        <property-ref ref="property4"/>
      </input-output>
    </item>
  </learning-object>
</environment>
```

The integration of QTI tests, information in SCORM packages or any other learning resources, can be achieved in a similar way. However, the different types of interfaces allow more fine-grained information handling. On the one hand, in many test situations there is no interest in supporting or distracting the learners with their previous answers if they have to retake a test or exam. On the other hand it is often helpful to the learners to access the results of a previous exam. In this case the learners should not be able to change the results. Using the officially published IMS-LD and IMS QTI integration, this cannot be achieved. The following fragment shows how the proposed interface layer would support this use-case.

```
<environment identifier="environment1">
  <title>First Learning Environment</title>
  <learning-object identifier="LO1" type="knowledge-object">
    <title>Examine your knowledge</title>
    <item identifier="item1" identifierref="QTIobject">
      <title>The Test</title>
      <output parameter="qtiparameter1">
        <property-ref ref="property1"/>
      </output>
      <output parameter="qtiparameter2">
        <property-ref ref="property2"/>
      </output>
    </item>
  </learning-object>
</environment>
```

```
<environment identifier="environment2">
  <title>Second Learning Environment</title>
  <learning-object identifier="LO2" type="knowledge-object">
    <title>Analyze the test</title>
    <item identifier="item2" identifierref="QTIOobject">
      <title>Test results</title>
      <input parameter="qtiparameter1">
        <property-ref ref="property1"/>
      </input>
      <input parameter="qtiparameter2">
        <property-ref ref="property2"/>
      </input>
    </item>
  </learning-object>
</environment>
```

The learning object in the first environment links the test provided in the QTI object to the properties “property1” and “property2”. In this environment these properties are connected only as output parameters. Consequently, the learner will see an empty test when the learning object is loaded as part of a learning activity. The second environment connects the same properties to the same test object. However, this time the properties are linked only as input. This means that the learner will not be able to change the test results.

In some settings it is necessary to inform a learning object about the environment in which it is made available. This is a typical use-case for widget applications that are used within a UoL. The proposed interface could serve as a solution for this use case. Instead of passing the value of a property to the item, the identifier of the environment is passed as a plain value to the item. This would allow the related item to setup the content appropriately for the environment. The following code fragment illustrates this solution.

```
<environment identifier="environment1">
  <title>First Learning Environment</title>
  <learning-object identifier="LO1" type="knowledge-object">
    <title>First Discussion Topic</title>
    <item identifier="item1" identifierref="widgetobject">
      <title>Discussion</title>
      <input parameter="environment">
        <value>environment1</value>
      </input>
    </item>
  </learning-object>
</environment>
<environment identifier="environment2">
  <title>Second Learning Environment</title>
  <learning-object identifier="LO2" type="knowledge-object">
    <title>Second Discussion Topic</title>
    <item identifier="item2" identifierref="widgetobject">
      <title>Discussion</title>
      <input parameter="environment">
        <value>environment2</value>
      </input>
    </item>
  </learning-object>
</environment>
```

4.4.2 [A.02] Flexible Service Integration

IMS-LD Level C notifications are meant only for one-way event-to-user communication. The related semantic constructs are tailored towards this task that other forms of notifications or immediate feedback loops cannot get modelled.

This has become a problem for modelling a closer integration of the MACE content federation services in the IMS-LD Level B UoL on Modern Architecture. The current solution provided by the UoL is to ask the learners to use the services. However, in professional education further contextualisation of such services is required for making Extensions and modifications of learning specifications and LMSs focused on adaptive learning

most efficient use of these services. Such contextualisation can be achieved by setting content related parameters, such as learner location, course keywords or competence levels.

Service integration is also highly important for modelling educational processes in interactive mobile and ubiquitous computing settings, as proposed in the UoL “Location based Information Support for Laboratory Visits”. Integrating ubiquitous computing services and mobile devices requires that information about the learning context is exposed to these services. As this could not get modelled with IMS-LD the non-interoperable workaround is discussed in Glahn and Specht (2010).

In addition, IMS-LD does not allow for many other service integration like, i.e. saving or retrieving data in external files, connection with external databases or modules developed with other languages are not described or supported within the specification (e.g. repositories).

There are two forms of service integration that are relevant for IMS-LD.

1. A service as part of a learning environment.
2. A service as a functional extension of the underlying background system.

The first case can get handled in the same way as interactive resources that were discussed in the previous section, through a communication dispatcher that allows for a bi-directional, live service integration.

The second case is only needed if the background service is explicitly required for a given UoL. In this case, an extension of IMS-LD Level C for supporting inter service communication is required. In addition, the integration of a “call-service” interface is suggested, so that it connects external services similarly to the notification semantics of the IMS-LD Level C version 1.0.

The main difference to the existing notification semantics will be that service integration is not based on unidirectional communication but requires also response handling.

This interface has to abstract several RPC interfaces such as SOAP, REST, or HTTP Query-string interfaces. Therefore, the modelling requires an abstraction from the actual service interface. The following example illustrates how this could be handled from the modelling perspective.

```

<if>
  <has-state state="finished">
    <learning-activity-ref ref="activity_1"/>
  </has-state>
</if>
<then>
  <call-service identifierref="SERVICE_ID" command="COMMAND_ID">
    <input parameter="PARAMETER_ID">
      <property-ref ref="PROPERTY_ID"/>
    </input>
    <output parameter="OTHER_PARAMETER_ID">
      <property-ref ref="OTHER_PROPERTY_ID"/>
    </output>
  </call-service>
</then>

```

This example calls the command that is defined as “COMMAND_ID” for the service defined as “SERVICE_ID”. This calls a service with the value of the property “PROPERTY_ID” as input. The response is (partially) translated into the property “OTHER_PROPERTY_ID”. In short, an information exchange through services is

proposed, making use of a heading (service) in combination with a set of parameters (variables).

The service interface can be provided as a separate resource in the content package. This resource will contain the rules for interface translation of a service, so it can be used by the “call-service” interface. For example, for connecting to SOAP services with WSDL descriptions, this resource would contain the WSDL. Such a service could be included into learning design’s resources section as following.

```
<resource identifier="SERVICE_ID" type="service" href="myservice.wsdl">
  <file href="myservice.wsdl"/>
</resource>
```

5 Connection between Grapple and IMS Learning Design

WP5 is mainly connected to WP3 and WP7. In general, the focus is on matching the needs for expressing adaptation in learning materials and processes (essentially Conceptual Adaptive Models (CAMs) created in WP3) within the capabilities and properties of existing specification frameworks like IMS-LD, possibly with extensions, so that these standards can be used within the Grapple framework (WP7).

The interaction between WP5 and WP3 results in an understanding of the various inputs, roles, interactions, adaptation methods & techniques which are feasible within the limits of the explored standards. The interaction between WP5 and WP7 results in the implementations done by WP7 of the theoretical LMS-based conversion models described by WP5.

Specifically, deliverable D5.3c will be mainly considered by the future deliverables expected in WP3 (D3.2c, D3.3c, D3.4c and D3.5c) and in the final release of the operational infrastructure in D7.5 (to be released in Year 3). Through these deliverables in WP3 and WP7, D5.3c will also impact the last cycle of evaluation actions in WP9 and WP10.

5.1 Connection to CAM and LAOS

As “D3.3a-Design of a CAM definition tool” and “D3.3.b- Initial Implementation of the Concept Adaptation Model Tool” show, the CAM model is based upon lessons learnt from the AHAM and LAOS (Cristea, A., & de Mooij, A.I., 2003) models, as well as being inspired by the multi-model, metadata-driven approach to content adaptation (De Bra et al., 2006), and has incorporated ideas from other models (i.e. Dexter, XAHM, Munich, UWE, and ADAPT). AHAM is a reference model for Adaptive Hypermedia Systems (AHS) which describes adaptive applications as consisting of three main layers: Domain model, User model and Adaptation model. On the other side, The LAOS model is an extension of AHAM. The Adaptation Model in AHAM has rules for updating the user model (e.g., with knowledge values), for defining aspects of the presentation (e.g. the presentation style for links depending on their suitability) and for domain-independent but only user-dependent aspects (e.g. a learning style). In LAOS, different aspects of the adaptation model are distributed over multiple layers in the model: generic and specific adaptation rules, and the presentation model.

LAOS is the closest model to Grapple’s, with a compare-able notation to IMS-LD. LAOS combines a model and authoring framework and IMS-LD is a specification (meaning an XML-based notation), so they cannot be easily compared. However, since “the structure of GRAPPLE authoring is a generalisation of the AHAM model, and either equivalent with, or

a generalisation of the more refined LAOS model” (D3.3a, pp.9), the connection between Grapple and IMS-LD can be defined through the LAOS model. Next, the high-level and low-level features similarities between the LAOS model and the IMS-LD specification are depicted.

5.2 LAOS' specifics

LAOS itself does not have a unique representation, unlike IMS-LD. It was created in order to represent *any* adaptive hypermedia authoring paradigm at the time. There are however several instantiations of its abstract, high-level ideas in the form of static content description, amongst which CAF was selected, for reasons as shown in the following subsection, and dynamic content description in various languages, including LAG.

Common Adaptation Format (CAF)

CAF is a portable XML format, extracting common and extraneous elements related to the way adaptive content is represented in most Adaptive Hypermedia authoring systems, and is used by popular academia systems such as AHA! (as input), MOT (as output) but also by commercial systems such as Content-e/LAOS (as output). CAF is a system-independent instantiation of the domain model and goal and constraints model in LAOS. In the GRAPPLE project, work has been carried out towards building a yet more generic version of CAF, to correspond to the multi-model vision. Still, CAF is an excellent tool for comparing the static part of the elements that take part in the adaptation process. Below, we show the Domain Type Definition (DTD) definition of the CAF file.

Layers of Adaptation Granularity (LAG)

The LAG language is an instantiation of the intermediate LAG model layer, the adaptation language layer. It allows for the creation of system-independent descriptions of adaptation. Finally, it also enables the creation of adaptation strategies, as defined by the LAG model. In the GRAPPLE project, LAG ideas and components are used to define an extended new language, the GAL (GRAPPLE Adaptation Language). Whilst GAL is still a work in progress, we can use LAG to compare the adaptation component in the authoring process of personalised education.

5.3 Comparison of high-level features of LAOS and IMS-LD

First of all, the high-level features of adaptive hypermedia authoring are examined, as described by the LAOS framework, and the e-Learning specification, IMS-LD.

5.3.1 General Content Representation

Comparison: From the point of view of representation and semantics, and especially, for general representation, IMS-LD enforces using an XML format to describe properties; and global XHTML files that use these properties for its content. Although AEH also allows XML representations (like CAF), the clear difference is that there is currently not one standardised way of describing the content, and different systems may use different ways of representing the same content. The use on XML and the introduction of IMS-LD as a standard makes it more portable and allows a high level of reuse. This is desirable for any authoring system for e-learning, and especially for authoring of adaptive material, which is notoriously complex and time consuming. For such authoring, the ‘write-once, use many’ paradigm is vital.

Conclusion: The field of AEH would benefit from a clearly defined and well thought through standard, and a unification of approaches. This is a clear issue for application of GRAPPLE in the close future.

5.3.2 Content extent

Comparison: Next, an IMS-LD manifest is significantly more verbose than the combination of CAF and LAG. A reason for this is that IMS-LD manifests need to specify much more information, which might or might not be relevant to the current application. This allows various enriched functionality, but costs in readability and space. Such enhanced functionality is illustrated by the following. After authoring in AEH, the author does not interact anymore with either the students or the content. This is mainly because the focus has been on creating automatic adaptation, and not on providing a software tool for teachers to communicate in real time with their students. In IMS-LD, provisions for such communication are, however, present. The latter therefore also has specific definitions of various roles, specified via the manifest.

Conclusion: This shows that IMS-LD and AEH have some complementarities. In terms of educational value of the experience, both IMS-LD, with focus on people and their roles in the learning process, as well as personalisation to the learner's needs, as supported by AEH, are necessary. The authors don't believe that a simplistic approach such as the extension of AEH with communication tools and definitions is desirable or necessary. It is conceivable that the two approaches could coexist together in learning systems, as it is explored in this WP5.

5.3.3 Generic conceptual point of view

Comparison: From a superficial generic conceptual point of view, both AEH and IMS-LD use a multi-layered method for describing the content and adaptation. However, a closer inspection shows that these are fundamentally different: in AEH, this is done via the authoring model layers, and in IMS-LD, via the different levels. However, the levels in AEH represent a clear separation between content, grouping of content and adaptation, whereas in IMS-LD, the division is based on certain functionality features. Past experiments (Cristea & Cristea, 2004) show that a clear separation of the adaptation from the content (such as in LAG) is very beneficial, as it allows re-use of advanced adaptation strategies created by programmers, for people with little or no programming knowledge (for example, non-technical teachers).

Conclusion: IMS-LD should allow for a clear separation of adaptation from content, as is supported by (some of) the adaptive hypermedia frameworks and methodology. This would allow a much more flexible approach to reuse of the authored products.

5.4 Comparison of low-level features of LAOS and IMS-LD

Next the low-level features or adaptive hypermedia (as described by the LAOS framework) and IMS-LD standard will be examined.

5.4.1 Static content representation

Comparison: Looking deeper into the data representation and semantics, the following was noted. 'Static' content (called 'domain content' in AEH) is represented by IMS-LD as XHTML documents, tagged as resources, and stored as separate files (see **Error!**

Reference source not found.) They can be both web-based contents and static learning resources (i.e. video, audio, text). In AEH, such content can be represented in various ways (there is no standard). CAF uses the domain concept hierarchy representation, storing all data for a lesson in one file. However, the AHA! adaptation engine (De Bra et al. 1998), for instance, interprets CAF data and divides it between several XHTML documents, in a fashion not that different from the IMS-LD representation.

Conclusion: XHTML representation of static content seems to be the best way to deal with the atomic, indivisible pieces of static information, that build the building blocks of an e-learning system, and that can be reused in various sequences and configurations to allow for personalisation to the learner.

5.4.2 Adaptation mechanism

Comparison: In this research a CAF file with a set of questions and their answers has been described, showing the latter only after learners have seen the questions. The way questions and answers are described is not very different in CAF or IMS-LD; yet, the way the adaptation is described differs greatly. For IMS-LD, rules are described via a hierarchical XML structure, and bound to a certain instance of the content, related to the Global Elements and definition of Properties, both used in the *ims-ld*-type files of a Unit of Learning Levels B and C. In LAG, rules are defined via a dedicated programming language, and can be reused as required. Most AEH represent adaptation at the level of adaptation assembly language (conform to the LAG framework), and formats vary: rules can be encapsulated into concepts, or kept separately, in XSLT sheets. For more complex amounts of content, allowing a similar adaptation strategy, the IMS-LD approach would definitely require more coding at an 'assembly language' level (as per the LAG framework definitions, whereas the higher semantics approach of LAG allows for reuse of the chosen strategy.

Conclusion: In order to obtain reuse, the *adaptation language* has to be separate from the *content description*. Adaptive hypermedia has already such separate languages (LAG, LAG-XLS) and is striving to create standard approaches to adaptation specification in GRAPPLE. Thus, another big difference between LAOS AEH and IMS-LD lies in the issue of reusability. AEH LAG strategies can be reused to adapt different content, as long as they are written in general terms rather than for specific concepts. In IMS-LD, reusability is focused on the learning flow of a Unit of Learning. Indeed, static content can always be reused. However, the main approach leans on the re-use of the learning strategy described in the manifest. Since the manifest needs Global Elements and Properties embedded into external xml files in order to interact with them, the reusability is possible, but highly demanding and, in the end, difficult to implement.

6 Conclusions and future research

This deliverable analyses the constraints of modelling with the IMS-LD specification (IMS-LD). We focus on adaptive learning processes. It translates concepts that have been raised in GRAPPLE into an interoperable educational design notation. This deliverable tackles the demand for integrating different educational techniques and technologies into coherent educational models. The conclusion is that IMS-LD will benefit from following the set of recommendations that has been provided in this report. They are categorised in two groups: Modelling and Architecture. Both categories lean on the principle of minimal modification. It means that the current version of IMS-LD (v1.0, 2003) is supported and Extensions and modifications of learning specifications and LMSs focused on adaptive learning

encourage its extension with new features and elements. In doing so, should these proposals be accepted and implemented by the standardisation body (IMS Global Consortium), the new version will not serve as a replacement of the current one, but an evolution. In addition, in specific cases, the modification of existing elements is suggested in order to improve the functionality of the specification. Nonetheless, these modifications are a minority.

The first group (Modelling) deals with adaptive learning processes, along with general modelling issues. It is based on the analysis and findings presented in D5.3a and D5.3b. It shows that a number of elements of the specification might be modified and-or extended to better support personalised learning. The second group (Architecture) concentrates on the need for a communication layer to allow for the information exchange between a Unit of Learning and an external learning resource (i.e. IMS QTI, SCORM, and others). In addition, the use of a service layer is proposed that can be used to integrate a Unit of Learning with already existing, external resources, like i.e. databases, repositories, and etcetera.

Forthcoming research concentrates on the proposal of new extensions and modifications like, e.g. interface adaptation through literate programming. In addition, it focuses on the actual promotion and implementation of these improvements through the appropriate channels.

On the other side, the precise extent both Grapple and IMS-LD can be connected still needs to be explored further. Since Grapple is a framework and IMS-LD is a specification, there are several layers in the Grapple's CAM that cannot be easily compared to the raw specification. Future research is necessary to find out how exactly the knowledge of the two fields could be combined; as IMS-LD is the de facto emerging standard, the best option might be to extend IMS-LD with the whole range of Grapple's functionality.

In addition, this deliverable compares the ontologies of IMS-LD and LAOS, as the closest, consolidated framework in which Grapple is based on, at a high level of semantics, as well as at a low level. This is an important step in connecting two seemingly unrelated fields, that of adaptive educational hypermedia and that of IMS Learning Design. For future research, it will be beneficial to experiment with concrete conversions, as the ones carried out with the LMS Sakai, Claroline, and Moodle.

7 References

7.1 Papers

ADL. (2000). Sharable Object Reference Model, SCORM. Retrieved May 9th, 2006, 2006, from <http://www.adlnet.org/index.cfm?fuseaction=Scormabt>

Burgos, D. (2008). Extension of the IMS Learning Design Specification based on Adaptation and Integration of Units of Learning (Extensión de la especificación IMS Learning Design desde la Adaptación y la Integración de Unidades de Aprendizaje). Doctoral thesis. University Carlos III, Leganés, Madrid, Spain

Burgos, D., Tattersall, C., & Koper, E. J. R. (2007b). Representing adaptive and adaptable Units of Learning. How to model personalized eLearning in IMS Learning Design. In B. Fernández Manjon, J. M. Sanchez Perez, J. A. Gómez Pulido, M. A. Vega Rodriguez & J. Bravo (Eds.), *Computers and Education: E-learning - from theory to practice*. Germany: Kluwer. Further info and download

- Burgos, D., Tattersall, C., & Koper, R. (2006). Re-purposing existing generic games and simulations for e-learning. Special issue on Education and pedagogy with Learning objects and Learning designs. Computers in Human Behavior
- Burgos, D., Tattersall, C., & Koper, R. (2007a). How to represent adaptation in eLearning with IMS Learning Design. Interactive Learning Environments, 15(2), 161-170
- Cristea, A. and Cristea, P., Evaluation of Adaptive Hypermedia Authoring Patterns during a Socrates Programme Class, International Journal Advanced Technology For Learning 1(2), ACTA Press, 2004
- Cristea, A.I., de Mooij, A., LAOS: Layered WWW AHS Authoring Model and their corresponding Algebraic Operators, WWW'03, The Twelfth International World Wide Web Conference, Alternate Track on Education, Budapest, Hungary 2003
- De Bra, P., Calvi, L. AHA! An open Adaptive Hypermedia Architecture. The New Review of Hypermedia and Multimedia, vol. 4, 115-139, Taylor Graham Publ., 1998.
- De Bra, P., Smits, D., Stash, N. (2006). The Design of AHA! , ACM Conference on Hypertext and Hypermedia, pp. 133, Odense, Denmark. 2006
- Glahn, C. & Specht, M. (submitted). Embedding Moodle into Ubiquitous Learning Environments. Submitted to mlearn 2010 conference.
- Gruber, M.R., Glahn, C., Specht, M., & Koper, R. (2010). Orchestrating Learning using Adaptive Educational Designs in IMS Learning Design. Paper to be presented at 5th ECTEL Conference, September, 2010, Barcelona, Spain.
- IMS Global Learning Consortium (2005). IMS Question and Test Interoperability Integration Guide, Version 2.0 Final Specification.
http://www.imsglobal.org/question/qti_v2p0/imsqti_intgv2p0.html
- Koper, R., Burgos, D. (2005) Developing advanced units of Learning using IMS Learning Design level B. International Journal on Advanced Technology for Learning (IJATL), Special Session on "Designing Learning Activities: From Content-based to Context-based Learning Services", volume 2, issue 3, October 2005
- Moreno-Ger, P., Burgos, D., Sierra, J. L., & Fernández-Manjón, B. (2007). An eLearning specification meets a game: authoring and integration with IMS Learning Design and <e-Adventure>. Proceedings of ISAGA. 2nd international workshop on Electronic Games and Personalized eLearning Processes (EGAEL2007). July, 9th-13th, 2007, Nijmegen, The Netherlands
- Tattersall, C., Burgos, D., Vogten, H., Martens, H.; Koper, R. (2006) How to use IMS Learning Design and SCORM 2004 together. Paper accepted for the SCORM 2006 conference [<http://ia.nknu.edu.tw/scorm2006/>]

7.2 Units of Learning

- Burgos, D. (2004-2010) Units of Learning and Learning Scenarios from #1 to #10 in the GRAPPLE Website at www.grapple-project.org
- Gruber, M.R., & Glahn, C. (2010a). IMS-LD Modern Architecture: Skyscrapers and Residential Homes Level A. IMS content package, Heerlen, The Netherlands.
<http://hdl.handle.net/1820/2550>. (UoL #11)



D5.3c - Extensions and modifications of learning specifications and LMSs focused on adaptive learning, v1-9-2011

Gruber, M.R., & Glahn, C. (2010b). IMS-LD Modern Architecture: Skyscrapers and Residential Homes Level B. IMS content package, Heerlen, The Netherlands. <http://hdl.handle.net/1820/2551>. (UoL #12)

Verpoorten, D., & Glahn, C. (2010). Dangerous Knowledge Tour on Web-usability. IMS content package, Heerlen, The Netherlands. (handle URL pending approval) . (UoL #13)