UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INGENIERÍA TELEMÁTICA

TESIS DOCTORAL

Orchestration of learning activities through the integration of third-party services in IMS Learning Design

Autor: **Luis de la Fuente Valentín**
Ingeniero de Telecomunicación

Director: **Abelardo Pardo Sánchez**
Doctor en Informática

Leganés, Abril de 2011

TESIS DOCTORAL

Orchestration of learning activities through the integration of third-party services in IMS Learning Design

Autor: Luis de la Fuente Valentín
Director: Abelardo Pardo Sánchez

Firma del Tribunal Calificador:

Firma:

Presidente

Vocal

Vocal

Vocal

Secretario

Calificación:

Leganés, ___ de _____ de _____

To my parents, who instilled in me the pursuit of knowledge
A mis padres, que me educaron en la búsqueda del conocimiento

To María, who's been the push I needed to reach my goals
A María, que ha sido el empujón que necesitaba para alcanzar mis metas

Agradecimientos

Teniendo en cuenta que *yo soy yo y mis circunstancias*, debo y deseo expresar el agradecimiento que siento hacia *mis circunstancias*, todas esas personas que de forma directa o indirecta han hecho posible esta tesis. Todo el trabajo abordado durante estos años (líneas de código, documentación, experimentación, divulgación...) no me pertenece únicamente a mí, a *mis circunstancias* también les corresponde su parte y lo justo es otorgársela.

Quiero empezar mencionando el buen hacer de mi tutor, Abelardo Pardo, durante todo este tiempo. Ha sabido dejarme libertad en las decisiones que han marcado mi línea de trabajo, pero también ha sabido detectar las ocasiones en las que mis pasos tomaban la dirección equivocada, de modo que sus certeros comentarios y consejos que me devolvían a un camino más acorde a lo que debe ser un trabajo de investigación. Si la pizarra de su despacho ha sido testigo de la creación de los primeros bocetos del trabajo ahora presentado, las sucesivas versiones han visto la luz gracias a las presentaciones y discusiones dentro del laboratorio *Gradient* al que pertenezco. Las reflexiones de Carlos, Mario, Jose Jesús, Raquel, Pedro, Derick, Mari Carmen y Maria Blanca han servido para pulir los detalles de mi trabajo, reforzando así la importancia del grupo incluso en algo tan personal como puede ser una tesis doctoral.

Los primeros pasos en un proceso nuevo son siempre complicados, más aún si este proceso viene con cambio de ciudad incluido. Por eso, es muy importante encontrar un grupo acogedor en el que sentirse a gusto y que ofrezca la motivación necesaria para que esos primeros pasos sean sólidos y estén bien orientados. Gracias a todos los que estábamos en el 4.1.A01 o similares: Dani, Pedro, Alberto, Jorge, Jose María, Juan, Jose Alberto, Natalia, Jose Carlos, Jonatan, Eduardo y en especial Jopez, que formaron entre todos ese entorno apropiado que te anima a seguir adelante y a no arrepentirte de tus decisiones. Han sido varios los despachos en los que he vivido durante estos años, y por tanto han sido varios los compañeros que han tenido que sufrir mis pensamientos en voz alta, mis quejas por no ser capaz de encontrar mis propios *bugs*, o mis quejas por culpa de los *bugs* ajenos. Gracias a Sergio y a Florina por soportarme en el 4.1.F05, y mención especial a Alberto, mi actual compañero en el 4.0.F06. Las innumerables ocasiones en las que entre los dos hemos arreglado el mundo, unas veces frente a una pizarra y otras frente a una cerveza, han sido al mismo tiempo vía de escape para los momentos difíciles y de inspiración en la búsqueda de soluciones. Muchas gracias Alberto, espero que termines pronto y con éxito.

Gracias a todos los miembros del Departamento de Ingeniería Telemática porque entre todos forman un entorno de trabajo que incentiva en lo profesional y agrada en lo personal. En especial quiero agradecer su paciencia a los profesores de Programación de Sistemas, que sufrieron en sus carnes la fase experimental de la tesis y todavía no me han retirado la palabra. El apoyo en los momentos de trabajo es importante, pero no tanto como el apoyo en los momentos de descanso. Gracias a Julio, Estrella, Derick, Pedro, Eduardo y Gustavo, mis compañeros de comedor más habituales, por hacer de la sobremesa un elemento imprescindible del día a día.

Por la naturaleza del trabajo presentado, la simbiosis con otros grupos de investigación

externos a la Universidad Carlos III ha sido necesaria y enriquecedora. La colaboración con el grupo GTI de la Universitat Pompeu Fabra ha producido resultados muy interesantes que han enriquecido esta tesis. Gracias a Mar y Davinia. También gracias al grupo de trabajo GSIC, de la Universidad de Valladolid, por su excelente capacidad colaborativa. Gracias a Asen, Yannis y demás miembros del grupo. Muchos de ellos me enseñaron durante mi etapa de alumno y lo siguen haciendo durante mi etapa investigadora.

Hay un grupo de personas que sin conocer las ideas de esta tesis son parte importante de ellas. Me refiero a mi grupo de amigos de Olombrada, que siempre están ahí para esa cervecita quitapenas y esas risas que hacen perder el estrés y la preocupación. Gracias a Enrique, Mirlo, Alfonso (los dos Alfonsos), Medardo, Ruth, Diana, Omara y todos los que siempre haceis de ese pequeño pueblo segoviano el mejor lugar del mundo.

Tampoco me olvido de mis compañeros de estudios en la Universidad de Valladolid, pero sobretodo amigos. En especial a Pablo, Jesús, Fabio, Richi, Mendi, Oscar y Loren. Sí, ya sé que siempre estoy agobiado y nunca tengo tiempo para quedar con vosotros. Prometo intentar organizarme mejor y poder quedar más a menudo, ya sea en Madrid, Olombrada, Zamora o Salamanca.

Cuando comencé la tesis, mi madre (sabia como todas las madres) me advirtió lo duro que es hacer un doctorado y yo no quise hacerle caso. ¡Cuántas veces durante estos años he recordado sus palabras! De mis padres he aprendido la inquietud por conocer, la necesidad de entender el por qué y el cómo de las cosas, y lo agradable que es la sensación de poder explicar. Mi hermana, Isabel, ha contribuido a ello y ha sido para mí un modelo a seguir en muchos aspectos de la vida. Gracias también a mi abuela Flores, que siempre está pendiente de mí y tiene más ganas que yo de que este doctorado termine de una vez por todas. Gracias también a la familia de María, que también es mi familia, por interesarse tanto por mí y por mi trabajo.

Y por último, María. Por estar ahí en los días malos, por hacer que los días buenos sean aún mejores, por demostrarme que el éxito es fruto del trabajo constante, por compartir mis alegrías y mis decepciones, por tu comprensión, por tu infinita paciencia, por tu perfeccionismo, por empujarme a ser mejor profesor y mejor investigador, por hacer de mí mejor persona, por todo lo que haces y también por lo que no haces. Gracias por todo eso y por muchas cosas más.

Resumen

La aplicación de las Tecnologías de la Información y la Comunicación al ámbito del aprendizaje tiene como resultado la ampliación del abanico de posibilidades en lo que a modelos pedagógicos se refiere. La aparición de lenguajes de modelado educativo permite la orquestación de actividades en entornos de educación a distancia. Esto hace posible la ejecución de cursos en los que priman la participación activa del sujeto y la interacción entre los diferentes actores del proceso de aprendizaje. La orquestación de cursos guiada por ordenador no es exclusiva de la educación a distancia. En escenarios presenciales, por ejemplo, puede suponer una importante reducción de las tareas administrativas del profesorado.

La especificación IMS Learning Design es el actual estándar *de facto* en el marco de los lenguajes de modelado educativo. Es frecuente la aparición de la especificación en las investigaciones más recientes, explorando su uso en el ámbito del trabajo colaborativo o en la creación de material adaptativo. Sin embargo, son varias las limitaciones que impiden una adopción práctica del esquema de trabajo propuesto por IMS Learning Design. Entre estas limitaciones, la falta de integración con herramientas de terceros dificulta la creación y el despliegue de cursos en los que el papel activo del alumno se refleje en el uso de herramientas basadas en la Web, especialmente en entornos de aprendizaje a distancia o semipresencial. Otro obstáculo importante es la falta de flexibilidad del modelo, ya que las herramientas de despliegue y ejecución de cursos se limitan a reproducir un guión previamente establecido, dejando escaso margen de actuación al profesorado.

Esta tesis caracteriza los problemas mencionados y propone una solución factible que no limite las características propias de la especificación, como son su interoperabilidad y expresividad. Para ello, se ha seguido una metodología de trabajo compuesta de tres fases: caracterización del problema, definición e implementación de la solución, y validación experimental del modelo propuesto.

Para la caracterización del problema se ha llevado a cabo un estudio del estado del arte con respecto a IMS Learning Design que se ha visto complementado con el diseño y despliegue de casos prácticos reales. En análisis de dichos casos prácticos se ha centrado en el estudio de los factores que afectan a las fases de autoría, despliegue y ejecución de los cursos. La documentación y posterior publicación de dichas experiencias supone por tanto una de las contribuciónes de esta tesis.

Tras la caracterización del problema, se propone una arquitectura que extiende la especificación IMS Learning Design. La arquitectura propuesta es independiente de la plataforma software que se utilice en el diseño y despliegue de cursos. Dicha arquitectura, que recibe el nombre de *Generic Service Integration*, permite la integración de herramientas de terceros en cursos guiados por IMS Learning Design. Esta integración se basa en la instanciación automática de herramientas externas y el intercambio de información entre las plataformas que intervienen en el curso. Así, se permite la inclusión de actividades que requieran el uso de herramientas basadas en la Web, sin que ello suponga una pérdida de las características propias de IMS Learning Design.

El modelo propuesto, *Generic Service Integration*, ha sido implementado como una ex-

tensión de GRAIL, el reproductor de IMS Learning Design en *.LRN*. Dicha implementación ha permitido la puesta en marcha de casos de estudio en los que la integración de herramientas ha sido un elemento primordial de la secuencia de actividades de aprendizaje. El análisis de dichas experiencias demuestra la viabilidad del modelo propuesto. Esta viabilidad se refiere tanto a la capacidad expresiva de la combinación de IMS LD con GSI, como a su alta replicabilidad y escalabilidad con un número alto de participantes.

Abstract

The range of applicable pedagogical models has increased with the adoption of the Information and Communication Technologies in the educational field. The so called educational modelling languages enable the orchestration of learning activities on distance education scenarios. It is possible, for example, to apply strategies that emphasise the relevance of an active participation of the subject and the interaction among the different actors of the learning process. Computer-mediated orchestration of learning courses can be extended beyong distance education scenarios to face-to-face experiences.

The IMS Learning Design specification is the *de facto* standard educational modelling language. The application of the specification in the support of collaborative learning models or in the creation of adaptive learning material is a frequent topic in current research. However, the model has several limitations that hinder the practical adoption of the IMS Learning Design framework. Among these limitations, the lack of integration with third-party tools is an obstacle for the creation and deployment of student-centred learning courses, where the active participation implies the use of Web based tools. Distance and blended learning models are especially affected by this limitation. Another factor that prevents full adoption of the framework is the lack of flexibility of the model: the existing players play a previously created script and leave no room for teachers' reaction to unexpected events.

This dissertation proposes a solution for the previous problems without limiting the intrinsic benefits of the specification, such as interoperability and expressiveness. The adopted research methodology consists of three phases: characterisation of the problem, design and implementation of the solution, and experimental validation of the proposed model.

The complete description of the problem has required a revision of the state of the art regarding IMS Learning Design and the design and deployment of several cases of study. The analysis of these cases has been centred in the study of the factors that affect the authoring, deployment and enactment phases of scripted learning courses. The documentation and publication of these experiences is one of the contributions of this dissertation.

An extension of the IMS Learning Design framework is proposed as a solution of the described problem. The extension, called *Generic Service Integration* is platform independent and allows the integration of third-party tools in courses described by IMS Learning Design.

The integration is enabled by the automation of administrative tasks such as the instantiation of external tools, and by the information exchange among the platforms that take part in the course. Thus, it is possible to include learning activities whose enactment requires the use of Web based tools without losing the intrinsic characteristics of IMS Learning Design.

The framework proposed by *Generic Service Integration* has been implemented as an extension of GRAIL, the IMS Learning Design player in the *.LRN* Learning Management System. Such extension has allowed the design and deployment of cases of study in which tool integration played an essential role in the sequence of activities. The analysis of these experiences demonstrates the feasibility of the proposed model. Such feasibility tackles two facts: first, the expresiveness of the combination of IMS LD and GSI; second, the replicability and scalability with a high number of participants.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

> I believe that scientific knowledge has fractal properties, that no matter how much we learn, whatever is left, however small it may seem, is just an infinitely complex as the whole was to start with.
>
> Isaac Asimov

1.1 Introduction

The active participation of the student in the learning process is recognised as one of the key factors for a successful learning. The idea of an active subject was considered by Piaget in his constructivist theory of learning [1]. According to him, the subject is responsible of the construction of his/her own learning through the experience and the active participation in the learning process. The social constructivism, proposed by Vigotsky [2], asserts the relevance of an active subject and contextualises it on a social environment. That is, the interaction with other subjects is an important part of the learner's construction of knowledge.

Both *constructivist* and *social constructivist* theories are reflected on the techniques used in formal education, where the learner is required to participate in the programmed activities. The *social* component emphasises the relevance of collaboration and cooperation among peers. Students' motivation plays an important role so that better participation is achieved with motivated participants. The pedagogical model supported by the social constructivism contains the following elements:

- Active participation of the student in the learning process. Knowledge acquisition requires the execution of practical activities. Thus, self-study is just another part of the learning process, and the emphasis should be put on the practical part of the course.

- Interaction with peers. Collaborative and cooperative activities reinforce the acquired learning and the positive interdependence helps on the achievement of transversal skills such as leadership or responsibility.

- Personalised learning environment. A high affinity with learning content increases the student's motivation. That is, the learner is more motivated the better adapted is the content to his/her needs.

The difficult management of all the factors that affect collaborative and adaptive activities hindered the adoption of such models. However, the relatively recent use of personal computers in the educational field enables the quick management of administrative tasks. Thus, the high requirements of constructivist based activities are relaxed so that they are more widely adopted and student-centred courses are offered to an ever increasing number of participants. Furthermore, the communication facilities offered by the Internet allows for the inclusion of interactive activities in distance courses, what was otherwise impossible to be enacted.

Although possible, the design of student-centred courses is a complex task due to the need of considering a large number of factors that may affect the overall result of the course. Course authors need to create learning material, design the activities and sequence all the elements so that the learning is maximised. The complexity of the creation process suggests the need of a method to reuse learning content in an effective way.

The idea of reusable learning material is not new. One of the very early attempts to promote reusability in the digital era was the definition of Learning Objects [3]. There are many different definitions of the term. Most these definitions describe the learning object as a piece of self-contained learning material that can be used in the creation of learning courses and that is easy to share and reuse. Despite the many institutional efforts aimed at the creation of public repositories, learning objects have not reached the expected level of use. A recent study [4] on the field reveals that larger learning objects are more reused than small ones. For example, a complete course about a certain topic is more reused than a single picture. According to this pattern of content reuse, some authors consider that learning objects cannot be separated from their context, because context is a crucial part of learning [5].

An alternative to face the reusability of learning material is offered by the so called *educational modelling languages*, which describe complete learning courses in such a way that they can be shared and reused. A course described by an educational modelling language usually consists in a sequence of learning activities, each of them defined by a description and a collection of related learning resources. The IMS Learning Design specification (IMS LD) is the most acknowledged existing modelling languages. It offers a framework to describe, deploy and enact learning courses in a reusable and interoperable way, at the time that supports a wide range of pedagogical learning models.

IMS LD is recognised by many authors as the *de facto* standard for course modelling [6]. According to the specification, a learning flow consists in a sequence of activities that are performed by different roles and that occur in a pre-defined context. The step-by-step delivery of activities supports the adaptation of the learning flow to the student needs, and also the creation of complex collaborative learning flows.

The three above mentioned characteristics of the constructivist pedagogical model (active subject, interaction with peers, personalisation of the environment) also describe social interactions that go beyond the learning scenario. The application of such ideas in the field of communication and information has resulted in what we know as the Web 2.0. In other words, the Web 2.0 is the technological support for the active interaction with peers in a personalised environment. The functionality of typical Web 2.0 tools is rather simple and clearly defined. Also, the interfaces are simple and agile. This design principle is oriented towards increasing the participation and number of interactions, because the added value of Web 2.0 tools comes from the number of users instead of the offered functionality [7, 8]. The breathtaking number of available tools grows rapidly and the adoption of Web 2.0

technologies have impacted the way we access the information in our everyday life, so that the user is now an active content producer instead of the old-fashioned passive content consumer.

How to reach such participation level on learning activities is one of the current challenges of student-centred pedagogical models. Therefore, current research is interested on the inclusion of Web 2.0 tools and techniques in learning courses. However, the distributed nature of the Web makes difficult to integrate its use in the centralised approach proposed by current learning methods and technologies: apart from designing interactions in a distributed environment, practitioners must consider how to track and evaluate students. In summary, it is not clear how the distributed resources and services offered by the Web can be orchestrated with a pedagogical sense and a reasonable administrative workload.

IMS Learning Design can provide support for the required orchestration. However, the framework does not provide mechanisms to integrate the so called *third-party tools* in the learning flow. Because of this limitation, courses whose activities are based on the use of the Web 2.0 are difficult to administer and replicate. Besides, the lack of integration reduces the adaptive capabilities of IMS LD courses due to the impossibility of using third-party tool's information to adapt the learning flow. That is, learning courses that use Web 2.0 tools have limitations on their reusability, self-containment, collaborative capabilities and adaptability.

The work presented in this dissertation analyses the tool integration problem and proposes a solution that has been implemented in an existing IMS LD player and validated with practical experiences. The proposal consists in a complementary framework for IMS LD. This framework enables the integration of third-party tools, which can be achieved without losing the principal characteristics of the original specification. The proposal, called *Generic Service Integration* (GSI), covers the complete course life-cycle and specifies the following requirements for authoring, deployment and enactment:

- During course authoring, offers a vocabulary to define *what* service is being included, *who* will use it and *how* it will be used.
- The deployment process is completed with the translation of the generic service description into a specific tool whose integration is supported by the corresponding service adapter.
- During the enactment, the users interact with the course resources and services while the IMS LD server exchanges information with the third-party tool.

The privacy issues derived from the batched information exchange between the two servers suggest the inclusion of a new phase in the course life-cycle. This phase, called pre-enactment, allows course participants to grant the permissions needed to request information from an external server.

The methodology adopted in this dissertation emphasises the relevance of the experimental work. In the first phase of the research work, the deployment and later analysis of field experiences allowed characterising the problem on its very detail. Later, the model was evaluated with the deployment of practical experiences that assessed the validity of the model. The experiences presented in this document demonstrate from a practical perspective the feasibility of the model and the benefits offered in large-scaled scenarios.

1.2 Objectives

The objective of this dissertation is:

To propose, implement and evaluate a framework that provides support for the integration of third-party services in the authoring, deployment and enactment of scripted courses represented in IMS Learning Design.

The accomplishment of this objective requires its division into the following specific objectives:

1. To deploy practical experiences that involves the use of IMS Learning Design in real courses. Such experiences will characterise the life-cycle of IMS LD courses and will also identify the benefits and drawbacks of the specification.

2. To identify the characteristics of IMS LD that should be also provided by an extension of the specification. The support for the identified characteristics would set the functional requirements of the framework proposed in this dissertation.

3. To identify the implications of third-party service integration in the course life-cycle of IMS LD scripted courses. The proposed framework will define the behaviour the supporting software in the authoring, deployment and enactment of learning courses.

4. To identify authentication issues of the interaction of IMS LD courses with the third-party services regarding the programmatic access to user's information. The analysis of the problem will consider security and privacy issues regarding the bidirectional exchange of information.

5. To propose a vocabulary to include third-party services during course authoring. The vocabulary will be conceived as an extension of the original IMS LD data model and will allow its inclusion in the course manifest with a minimum impact on the rest of the course definition.

6. To propose an architecture that enables the implementation of the proposed framework in existing IMS LD players. The architecture will be pluggable with service adapters, that provide support for specific third-party tools. The model will enable quick development of new service adapters.

7. To provide an implementation of the proposed framework in an existing player. This specific objective includes the development of service adapters that validate the model and enable experimental usage of the software.

8. To deploy practical experiences of use where the model can be validated in real situations. The analysis of such experiences will be focused on assessing to what extent the requirements identified in the specific objective number 2 are accomplished.

1.3 Research methodology

The work done in this dissertation belongs to the field of technology enhanced learning. In particular, it focuses on making technology useful in learning scenarios. The main objective is the definition of a framework that enables the integration of third-party tools in scripted learning courses with a flexible enactment. The research is, therefore, an engineering research and the used methodology must be understood as such. According to [9], engineering research methodology (which has been followed in the presented work) is composed of the following four steps:

Information phase. The goal of this first step is to identify the existing characteristics of the problem domain and to clearly state the subject under research. This phase usually consist in the revision of the existing literature. In this dissertation, the information has been gathered from the following sources:

- The review of the literature provided a theoretical background of the problem domain and the existing work in the field.
- The identification of research groups working in similar problems enriched the discussions of the matter, with the participation in workshops, and the development of coordinated field experiences.
- The few number of documented practical cases of study in the research field suggested the development of experiences that contributed to the literature with empiric knowledge of the problem domain.

Definition phase. The information gathered from the previous phase results in the definition of a solution that overcomes the limitations presented in the existing alternatives. In this dissertation, such solution consists in a framework that enables the integration of third-party tools in scripted learning courses and provides flexibility during the course enactment. The definition of the proposal emphasises its interoperability, allowing its implementation in any exiting IMS LD player.

Implementation phase. The implementation of the proposal assesses its practical feasibility and allows the deployment of case studies oriented towards the validation of the proposed model. The framework proposed in this dissertation has been implemented in GRAIL [10] and it has been published under open source license.

Validation phase. The last step of the applied methodology is the definition and deployment of experiments that evaluate the validity of the proposal, in order to show and document how the proposed solution overcomes the limitations identified in the information phase.

The experimental phase of this dissertation consisted in the deployment and later study of three cases of study that used the proposed framework in different scenarios. The experiences were analysed with a mixed evaluation method [11] that combines qualitative and quantitative analysis of the results.

1.4 Structure of this document

After this introduction, Chapter 2 provides an overview of the use of Web technologies for educational purposes. The goal of the chapter is to provide a background on the history, current situation, and trends of the research area. Such overview contextualises the concept of scripted courses and helps on the understanding of the need of tool integration. The chapter also presents the standardisation efforts in the field of *e-learning* and the most successful initiatives.

The third Chapter focuses on the state of the art regarding IMS Learning Design specification. First, the need of scripted orchestration methods is discussed with the definition of instructional design techniques and their application in student-centred learning paradigms. Then, the characteristics of IMS LD and its supporting software are presented. The chapter finalises with a review of the most relevant uses of IMS LD, with special focus on the service integration problem.

Chapter 4 details the software platform that hosted the implementation and experiences done in this dissertation. The chapter offers some background of the decissions that resulted

in the implementation of GRAIL and the election of the .LRN platform. The technical description of the software is completed with a description of the *flexibility* concept and the features offered by GRAIL oriented towards improving the flexibility of learning courses.

Generic Service Integration is presented in Chapter 5 as a framework that provides IMS LD with support of the integration of third-party tools. First, the proposal is presented including the defined architecture and its interaction with the IMS LD framework. Then, the impact of GSI in the course life-cycle is discussed. Identity and authentication issues are an important element of the proposed framework, so the problem statement and the proposed solution are detailed in the text. Finally, the chapter presents the details of the implementation developed as part of GRAIL.

The experiences presented in Chapter 6 correspond to the first part of the methodology: the information phase. The described experiences were taken in prior to the definition of GSI and provided expertise in the field. There are three presented experiences: the first one regards course authoring and the expressiveness of the specification; the second one focuses on the deployment and enactment of complex collaborative learning flows in distance scenarios; finally, the third experience is the development of a proof of concept that relates orchestration of learning activities with a finer grained orchestration provided by specialised tools.

The experiences that evaluate the validity of the proposed framework are presented in Chapter 7. The experiences are three cases of study aimed at the evaluation of specific characteristics of GSI. The first case of study analyses the technical feasibility of the approach and how the instructors understands the model. The second experience studies the support offered by GSI in the deployment of traditional learning models. The support of innovative scenarios that uses different technologies in different spatial locations is analysed in the last presented experience.

Finally, Chapter 8 draws the conclusions of the dissertation, summarises the most relevant contributions, and state guidelines for future research in the field.

Chapter 2

Web Based Tools in the context of *e-learning*

> Web 2.0 is not a technological revolution, it is a social revolution
>
> Stephen Downes

2.1 Introduction

The wide adoption of the World Wide Web caused a shift in the way we exchange information. With the Web, people can instantly access to an ever increasing catalogue of content, but they can also produce this content or interact with peers with sophisticated communication tools. Such communication can be synchronous or asynchronous, text or multimedia based, etc. The large number of possible combinations results in innovative applications in very different fields.

In education, the use of Web technologies to scaffold learning courses is usually referred as *e-learning*. The boundaries of what is and what is not *e-learning* are not easily defined since *e-learning* can range from self-paced online tutorials to instructor-led graduate courses at a university. The definition of *e-learning* is not limited by specific technologies or current delivery systems [12]. What is clear is that the availability of new technologies is reflected in their educational applications.

In the very beginning of the Web, the difficulty of publishing content leaded *e-learning* to a content-producer/content-consumer paradigm, where the material was created and published by educational institutions and the learners received and studied such content. With Web 2.0 tools, the learners are enabled to produce and to interact with peers. In order to produce effective learning, such interactions require to be orchestrated so that the activities gain a pedagogical sense.

The active participation of the learners and the personalisation capabilities of the Web 2.0 lead *e-learning* to a scenario where it is the learner who decides what to learn and how to learn it. Following this trend, the *Personal Learning Environment* (PLE) is a rising idea that is gaining acceptance among *e-learning* researchers [13, 14]. The PLE is an environment where the user has a total control of the learning process and all the tools, activities, communities of practice, etc. are chosen by the learner itself. Still under debate, the idea of a PLE shows the direction of future *e-learning* developments [15].

The evolution of *e-learning* has stimulated the standardisation bodies in the definition of models that provide enough expressiveness to satisfy the need of educational models and promote interoperability of learning courses. The goal is to create a widely accepted framework that enables an efficient exchange of *e-learning* material. As an example, the SCORM model is widely adopted to deploy courses in learning platforms. However, standardisation is difficult in a scenario where new functionalities are continuously appearing.

This dissertation explores and proposes orchestration methods for Web 2.0 based activities. It is therefore needed, in order to understand the *e-learning* trends, to provide some background on the evolution of *e-learning* technologies and their impact on the educational field. This chapter summarises some of the most relevant topics in the area and it is organised as follows: first, the evolution of Web based learning technologies is presented in Section 2.2. The concept of Personal Learning Environment is presented in Section 2.3 to exemplify current *e-learning* trends. Finally, Section 2.4 provides an overview of the existing standardisation efforts in the field of *e-learning*.

2.2 The Web and the learning process

Since the very early years of the Web, it has been used as a mean to deliver content in distance situations. The technical skills required to publish content led the learning usage of the Web to follow a content-producer/content-consumer paradigm. One step forward was the development of Learning Management Systems, that offer easier methods to create content. The so called Web 2.0 allows all users to produce, publish, receive and give feedback.

Educational models take advantage of the existing Web tools and allow new interacting methods among course participants. The path opened by this new learning paradigm is still to be explored, but the initial results are promising. This section presents how the Web is being used in education and where the research is leading us.

2.2.1 Learning Management Systems

The management of information in learning courses involves certain tasks that can be accomplished into what is usually called a *Learning Management System* (LMS). A LMS is a software that centralises the administration and delivery of learning courses. This type of systems is referred in the literature as *Content Management Systems*, *Learning Content Management Systems*, *Virtual Learning Environments*, *Course Management Systems* and so on. All these systems essentially share the same functionality and it is difficult to classify them. In this text they are referred as LMS.

There is a wide market of proprietary and open-source solutions. Well known alternatives are WebCT, Blackboard, Moodle or Sakai. Some examples of the complete list, which is much wider than that, can be found at [16]. The 80% of the current market (at least in the United States) is dominated by WebCT, Blackboard and Moodle [17, 18]. According to [19], LMS can be benchmarked with following parameters:

- Institutional support
- Course development
- Teaching and learning
- Course structure
- Student support
- Faculty support

- Evaluation and assessment

Educational institutions have widely adopted LMSs to deliver distance and blended courses. These platforms are also used to improve the learning experience in traditional courses. Actually, LMSs are used three times more often for technology-enhanced traditional courses than for online courses [20].

A characteristic in LMSs is the centralisation of tools and functionalities in a single platform. This fact prevents learners and instructors to use their preferred Web 2.0 tools to interact with content. For example, they cannot use Flickr to share images because the LMS provides a tool with similar functionality. Next subsection explains that the very potential of Web 2.0 lies in the amount of users that use the same tool instead on the provided functionality.

2.2.2 Web 2.0 tools and *e-learning*

According to the Bloom's Taxonomy [21], knowledge is acquired at different incremental levels. Learning at a given level implies the acquisition of the knowledge from previous levels. The model proposed by Bloom allows instructors designing more effective material oriented towards the achievement of certain objectives. Each of the six defined levels (knowledge, comprehension, application, analysis, synthesis and evaluation) can be reached with different activities. For example, the analysis skill can be acquired through the creation of conceptual mindmaps or writing an abstract.

The Bloom's Taxonomy, proposed in 1956, is far from being an obsolete model. The arrival of new tools increases the number of available learning activities and it is possible to establish a relationship among these new activities and the different levels of the taxonomy. This is precisely the objective of [22], which analyses to which level corresponds each of the tools of the digital era.

The availability of new tools means more than a complement to the existing ones. The Web 2.0 has impacted our everyday life in the same way as it is doing with education. The work presented by [7, 8] analyses what the big ideas of Web 2.0 are, which can be summarised as follows:

- The most successful tools are those whose functionality is rather simple and that has been clearly defined. For example: *Flickr* stores photos and allows sharing them.
- User interfaces are simple and, more important, agile. Web based tools are as fast (if good Internet connection is available) as desktop tools, but do not need to be installed.
- A service is more useful the more users it has. Thus, there is no emphasis on better functionality aimed at the individuals, but on a better usage of users' contributions so the community can perceive the added value.

Internet users have reacted to the new technology with a shift on the demand. The old Web was only readable, Web 2.0 is a *read-write Web* and, where content was expected, now users ask for the chance of participation. The passive attitude of content readers has turned into an active behaviour where users want to publish their own content, comment on other's and receive instant comments. In the Web 2.0, where reactions are immediate, the community is the content producer and the large amount of available tools is the platform that supports the interactions.

The ideas of the Web 2.0 fit in the constructivist theories: the knowledge is not transferred; it is built inside the learners' brain thanks to the interactions with the content and other learners. That is, the *e-learning* (mostly based on content delivery and consumption) has an opportunity to become *e-learning 2.0* [23].

As analysed in [24], the use of Web 2.0 tools in the context of *e-learning* has well stated pedagogical principles. Current research is very active in the exploration of the impact of Web 2.0 in learning methods. An example of such abundance is [25], that applies wikis and blogs in the scaffolding of a problem based learning activity, or [26], that identifies the main characteristics of wikis, blogs and forums and assess the implications of those characteristics. Another example is the work presented in [27], that analyses the impact on learning and cognitive style of using blogs and podcasts on a programming course.

Web 2.0 suffers a constant improvement and the most recent applications are waiting to be incorporated into learning models. Despite all the effort on understanding how *e-learning* 2.0 can be applied in real education, there still are important drawbacks to overcome and more research is required in the field. One of the identified drawbacks is described as follows: the emphasis of self-guided activity promoted by the Web 2.0, but studies have shown that students rarely develop explicit learning strategies on their own [28] and there is still the need of tutors to make the most of the learning process. That is, there is a need for an orchestration method for the sequentiation of Web 2.0 based activities.

2.2.3 Social Networks

The interactive nature of the Web 2.0 the base of its success. Interaction appear at different levels: first, the learners interact with the learning material; second, the learners interact among themselves. This social interaction is one of the most relevant factors of the widespread adoption of Web 2.0 tools. Social software, which can be broadly defined as tools and environments that supports activities in digital social networks [29], allows Internet users to easily form communities and share common interests.

The membership in social networks is self-selected (the user decides if he/she enters the network or not), and the network is self-organised. The educational application of social networks suggests that learners can lead their own learning interests and that there are easy networking methods that allow to share experiences, knowledge and activities in an effective way. This is, therefore, an active matter of research, as shown in the literature review presented in [30, 31, 32].

Learning communities are enriched by the participation of the individuals. However, the interaction is not always as spontaneous as it could be desired. As explained in [29] individuals participation can be motivated by four major reasons: *personal access*, the information offered to a user is more personalised the more active the user behaves; *personal reputation*, when the user can increase his/her influence and visibility on the community; *social altruism*, if the user perceives his/her contributions as a 'public good'; *tangible rewards*, when the user's participation is motivated by the possibility of obtaining a prize, high grades in the case of formal learning.

With social software, it is easy to form teams whose members will collaborate to develop a task. A common problem in education is how to select the best peers for each team. Social networks hold large amounts of information that can be used for teams' formation. Just as an example of current proposals, the works presented in [33, 34] are based on agents running in background to detect subjects the user is interested in. A deeper study on the field can be found at [30].

Social software offers the opportunity to enrich learning activities in several ways. However, it also presents some drawbacks that hinder the effective use of Web 2.0 tools in *e-leanring*. For example, the more tools they use, the more difficult to track students interactions. It is very common, in studies of learners interactions during a course, to find that the authors states that "the number of entries could not truly reflect the activeness of a group, which might prefer other means of communication" [35]. Due to the large amount of existing alternatives for an effective interaction, the difficulty of tracking students' interac-

tions is a problem that affects the evaluation process. It is needed a trade-off between free interactions and traceability of the process, which is in any case, difficult to achieve.

Another problem is the lack of reusability of Web 2.0 based courses. That is, communities are willing to participate in new activities or discussions, but they usually refuse to repeat already performed ones. When a certain topic has already been discussed in a forum, any attempt to reopen the discussion will be redirected to the archived posts. However, a better learning would require the learners to be involved in the discussion, instead accessing the final result. A hypothetical course that requires the interaction with experts in a field will be difficult to be reused. This fact hinders the reusability (whose relevance is discussed in Section 2.4.1) of learning activities based on the interaction with already established communities. However, this is not always true: the expected interaction level has been successfully gathered in subjects such as second language learning [24]. This suggests that the reusability of courseware is activity and field dependant.

The application of social software in the design and enactment of learning activities is a promising method in the Computer Supported Collaborative Learning (CSCL) field of study.

2.2.4 Virtual worlds

The interaction-among-peers principle that prevails in the Web 2.0 is also present in 3D virtual worlds, such as Second Life [36]. Represented by avatars, which usually are human-shaped animations, users interactions imitate real world ones, allowing more creativity, freedom and innovation, and are fertile ground for all human endeavours (socialisation, recreation, creative/artistic expression, commerce, and learning) [37]. Virtual worlds are also immersive worlds. That is, users are aware of their presence in the world and this fact impacts on the affective, empathic and motivational aspects of the experience.

Virtual worlds provide a shared, realistic, and immersive space where users, by means of their avatars can explore, interact, and modify the world, in addition to communicate and collaborate with other users in both synchronous and asynchronous ways [38]. The ecosystems hosted by virtual worlds have several characteristics that make them suitable to develop effective learning experiences:

First, they enable learning experiences to be contextualized. The situated learning approach [39] can be deployed in virtual worlds. For example, in [40] the learning is supported by virtual field trips that allow learners to visualise the process and the result of building production plants. Those trips would not be possible otherwise because of their high cost, but are easy to perform in virtual worlds.

Simulations are a different perspective of field trips. While the real world is limited by the laws of Physics, virtual worlds can break the rules: field trips into the human body, changes in gravity parameters and so on. Experimentation through simulation can be enriched with virtual worlds

Another interesting characteristic for learning is that 3D virtual worlds allow the exposure to the authentic culture. For example, [38] uses Wonderland [41] to recreate emblematic places of Madrid in order to contextualize the learning of Spanish as a second language. This work also exploits communication characteristics of virtual worlds, so the learners train writing, reading, listening and speaking skills by interacting with the different elements of the world and with peers.

It is also interesting to explore the interaction between the real and the virtual worlds. In the experience presented in [40], a lecture in the University of Hamburg can be attended both in the classroom and in Second Life. The configuration of the real and virtual lecture was in such a way that all students were aware of the presence of the 'other world', and

interaction among all participants was enabled. This type of settings opens the door for new methods of distance education.

Unlike *massively multiplayer online games*, avatars in virtual worlds such as Second Life are not assuming any role and they do not have a clearly defined task to perform. The consequence is that users have to build their own history. The good view is that learners can lead their own learning, as dictated by constructivist theories. On the contrary, users usually felt lost without an objective to accomplish and they left the world. Instructional design techniques would be required in order to engage participants in the activity and thus achieving narrative immersion.

2.3 Personal Learning Environments

Due to the high adoption level of LMS, it can be said that this is the dominant design in Web based learning. The LMS is powerful for formal, institutionally-mediated learning. However, they are too rigid for informal learning, whose intrinsic characteristics suggest a more flexible and personalizable environment.

In the context of informal learning, a relatively new concept is gaining the attention of the researchers' community. This concept is the *Personal Learning Environment* (PLE). The big idea around the PLE is to allow learners to customise their environments to their finer grain details so they can learn in a self-created space where they choose the topics of interest.

What is a PLE is still under debate. It is commonly agreed that the PLE is not a software, but requires software support. Most of the approaches to the concept are related to the use of Web 2.0 tools. Due to its relationship with the Web, this section introduces the concept of PLEs and how current research is facing its prototyping.

2.3.1 History

According to [15], the first approach to the concept without mentioning its name is the *Future Learning Environment*, a work presented in 1998 by the Media Lab in Helsinki, which is a server based environment that supports learner and group centred work. The PLE term was first mentioned and discussed in an unpublished work [42] of the JISC/CETIS centre.

The term acquired some relevance and it featured as session topic in the JISC/CETIS conference in 2004. The development of prototypes for the idea started a year later and the earlier publications discussing the PLE concept appeared in 2006 [13, 14].

At the date of the writing of this dissertation, ideas about PLEs are still forming, and the literature reveals an increasing interest on the concept, while future guidelines are still not clear: some works claims that more theoretical discussion is required to set the basis of the proposal and the requirements of an implementation [15], while other initiatives consider the concept mature enough to be related with existing technologies such as ubiquitous computing [43].

2.3.2 Characteristics

The development of Personal Learning Environments is motivated by the needs of life-long learners to operate with systems that are under the control of the learners themselves, and not by institutions, while personal information and configuration can be maintained across institutions. Therefore, the construction of PLEs is focused on the acquisition of learner-centred systems that can be connected with other -institutional or not- systems.

The definition of Atwell [44] is related to lifelong learning, social learning and informal learning. A PLE is defined as "a technological environment constructed by an individual and used in everyday life". The PLE is not restricted to be used in front of the computer and can take advantage of ubiquitous computing, via mobiles phones for example. Furthermore, PLEs can be used in different contexts, allowing to bring together various types of learning such as self-motivated learning, school learning, workplace learning and so on. In short, it can embrace all formal and informal learning [15].

The concept of PLE is usually introduced as an alternative of LMSs. However, these two systems are not necessarily incompatible. Both systems can live together and complement each other. The following text summarises the description of the PLE given at [13], where the comparation with LMSs is used as a vehicle to introduce the ideas around the PLE. Table 2.1 depicts a summary of these ideas, explained as follows:

Use of tools. The PLE coordinates the use of several tools in a single environment, while these tools are used in their context. Instead of using tools from a single provider, learners can select their preferred tools and create a context to use them. In LMSs, tools and content are institutionally integrated as part of a course.

Permissions. According to the Web 2.0 principles, the verbs in the PLE are "consume" and "produce". All learners can participate in activities and coordinate them, because all of them have the same permissions. Such horizontal distribution of participants does not appear in LMSs, where teachers have privileges over students.

Interface. While the LMS offers the same interface (apart from personalisation options) to all learners, the PLE is fully self-configured. Each learner is different and has a different view of the learning environment. Learners are responsible of their learning and also of their environment.

Tools Integration. From an institutional perspective, integration with third-party tools requires the use of open standards that ensures interoperability. However, such requisite reduces the number of integrable tools and thus the number of potential users. PLEs use open standards, but do not close the door to proprietary APIs.

Rights. Sharing content is sharing knowledge. Learners consume and produce open and unstructured content which does not fit with formalism such as learning objects. In the PLE, learners can create playlists of their preferred content and share them, as in the Web 2.0.

Scope. What happens in the LMS, stays in the LMS. Institutional systems are rarely connected with other platforms and the interactions that occur in such platforms have a limited scope in terms of time and repercussion. On the contrary, the activity in the PLE is open to Web 2.0 tools and the interactions may occur with any other participant in the world, and can be revised time after they have concluded.

All above definitions draw the PLE as software tool. However, the term "environment" reflects that the PLE goes beyond the software and states about ubiquitous computing, ubiquitous learning. The content of the PLE grow with the learner activity, no matter if this activity in is a computer mediated task of an activity of the physic world.

Since the first use of the term until now, there is no formal definition for PLEs and the subject is still a debate. What is clear is that PLEs allow learners to select their topics of interest, sources of information, tools to access and create content and, in short, take the control of their own learning in an active manner. PLEs are therefore tools enabled to support constructivist theories of learning.

Learning management System	**Personal Learning Environment**
Tools and content integration within a course	Tools to connect user and content
Asymmetric relationships	Symmetric relationships
Homogeneous experience	Individualised context
Open standards	Open standards and proprietary APIs
Access control	Open content
Organisational scope	Personal and global scope

Table 2.1: Characteristics of the PLE as an alternative to the LMS.

2.3.3 Adoption

The PLE is a very promising concept, but it is still a concept. In other words, most applications are prototypes that allow to test the concept and develop experiences of use, but not really ready to be adopted by real users. A good compilation of PLE architectures is presented in the Edutech Wiki [45].

There exists tools such as Colloquia [46] and Fle3 [47] constructed upon a learner-centred or group-centred approach and somehow follow the principles of the PLE. Both tools have been used by real users but, in both cases, their development was made before the term PLE was introduced.

The literature shows an upcoming interest on the creation of PLE. For example, PLEX is a desktop tool developed at CETIS that allows managing people, activities and resources. The work presented in [43] explores how ubiquitous computing is related with this type of learning environments, but no actual prototype is presented. Most of the ongoing implementation establishes a relationship between the Web 2.0 and PLEs [48, 49, 50].

Mash ups

The Web 2.0 and the PLE meet in the so called *mash ups*, which are compositions of already available tools that provide more functionality that the single use of the composing tools.

In mash ups, services are integrated by their functionality. That is, the output of a service is used as the input of another one. There are thousands of available Web mash ups following this approach [51], some of them focused on education. There exist also *mash up factories* such as Yahoo Pipes, that allow end users to build their own mash ups.

From a different perspective, services can be aggregated and accessed from a common interface, being their functionality comprised in widgets. Examples of such approach are iGoogle, Netvibes and Pageflakes.

From an educational perspective, mash up factories and widget-based mashups allow learners to easily construct an interface composed by their frequently used tools. As argued in [52], mash ups are a promising approach to construct PLEs. The relevance and upcoming interest of mash ups in education is evident with the recent appearance of dedicated special issues in journals [53] and workshops in conferences [54].

2.4 *e-learning* Standards

Learning technologies are quickly evolving and the number of available tools is also rapidly growing. In an emergent market such as the Web 2.0, vendors attempt to acquired the leading position and, therefore, make their own format to be considered as the *de facto*

standard. In such scenario, reusability of courseware, when possible, has a great cost. This section analyses the existing groups and initiatives working for the standardisation of *e-learning* formats and methods.

2.4.1 Reusability

Functionalities offered by learning management systems go beyond the educational purpose and are also administrative tools integrated in the institutions. On the other hand, Sections 2.2 and 2.3 state that the trend in learning systems is the distribution of the functionality among different providers, specialised in a specific tool.

Such amount of learning tools suggests the need of reusability of content. Otherwise, effective courseware would only be able to be deployed in a very specific platform and the spectrum of potential users is then reduced. Furthermore, information exchange among different vendors requires the agreement of a common format or a big expense in the adaptation of the material. In order to maximise effectiveness, high quality courseware is worth to be reused. The concept of learning object aims at the composition of courseware through the composition of smaller, effective parts. The lack of a common format to store learning objects is a barrier for the productivity of courseware creation.

The goal of standardisation organisms is to reach agreement among vendors in common formats and methods to produce and deliver *e-learning*, so the reusability and interoperability of courseware is maximised and the benefit of *e-learning* systems can be increased. The relevance of standards has been argued in [55, 56].

2.4.2 Specifications and standards

Sometimes misused and confused, it is important to note the difference between a standard and a specification. ISO/IEC Guide 2:1996, definition 3.2 defines a standard as "A document established by consensus and approved by a recognised body that provides for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context". The specification, on the contrary, has not been ratified by any organism of institution and, with enough support of institutions; it is used in a provisional way.

A specification is created in order to overcome a detected problem in a given context. When the specification is formalised with the corresponding document, it starts being used and the activity around it results in new revisions of the specification so it is better adjusted to the real scenario. Then, standardisation bodies ratify the specification and then the proposal becomes a standard (Figure 2.1).

The standardisation process is long. Meanwhile, specifications are used and, when the gain enough recognition, usually referred as *de facto* standards or simply standards. Currently there are few recognised standards in the field of *e-learning*, but there is an intense activity in organisations that develop *e-learning* specifications.

2.4.3 IMS Global Learning Consortium

The IMS Global Learning Consortium (IMS) appeared in 1995 with focus on higher education scenarios. However, all the specifications published since then have a general scope and can be applied in a wide range of contexts. The consortium is a non-profit organisation oriented towards a worldwide adoption of learning technologies through the publication of specifications that promote interoperability among systems and reusability of content.

The acronym *IMS* originally stood for Instructional Management Systems, but over time the consortium realized that these terms did not reflect the actual field of study of the

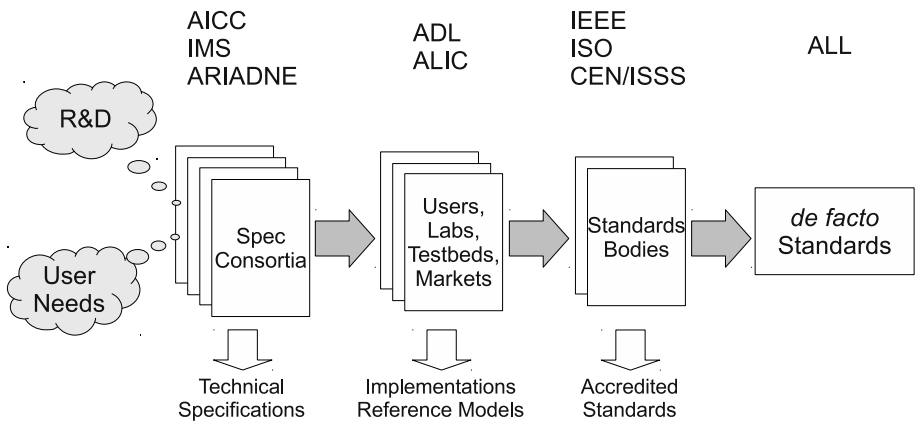


Figure 2.1: A model for standards evolution.

initiative. As a result, they recommend to be referred as IMS Global Learning Consortium, or simply IMS, without using the complete name.

The IMS is supported by more than 190 organisations that come from different sectors such as educational institutions, software vendors, content providers, government agencies and other consortia.

Since its creation, a large number of specifications have been published under the umbrella of IMS, and all of them can be accessed through the Web [57]. The specifications proposed by the IMS cover a wide range of aspects in the *e-learning* and have a great impact on the research community. Many of the IMS specifications have been adopted as the *de facto* standard (IMS CP, IMS LD, IMS QTI) and they are often used as part of other recommendations as in the case of SCORM. The IMS consortium is currently pushing the use of Common Cartdrige as a more powerful alternative for SCORM. An overview of current IMS specifications can be viewed in Table 2.2.

2.4.4 Aviation Industry Computer-Based Training Committee

The Aviation Industry Computer-Based Training Committee (AICC) [58] was created in 1988 and provides guidelines for the production and delivery of computer-based training. The guidelines offered by this organisation have a general purpose, so they are not limited to be used in aviation.

The most relevant of the AICC specifications, AICC/CMI, provides interoperability for *Computer Managed Instruction Systems*. It was first developed in 1993 for CD-delivered courseware and has been updated in 1998 and 1999 so that it is now oriented to HTTP-based courses. The last release includes a JavaScript API that allows courseware to interact with the learning management system. Some of the concepts defined by CMI were used for the SCORM specification, and it is the latter which has received more attention but, in several aspects, CMI and SCORM are very similar.

IMS specification	**Date**	**Brief description**
IMS Basic Learning Tools Interoperability	May, 2010	Allow remote tools and content to be integrated in a LMS
IMS Access For All	May, 2010	Meet the needs of learners with disabilities and anyone who is disabled by their context
IMS GLC Common Cartridge	Oct, 2008	Package and describe learning material (content, assessment, rights, etc.)
IMS Meta-data	Aug, 2006	Defines the attributes required to describe learning objects
IMS Tools Interoperability Guidelines	Mar, 2006	Allow remote tools and content to be integrated in a LMS
IMS General Web Services	Jan, 2006	Promotes interoperability for Web service based specification implementation.
IMS ePortfolio	Jul, 2005	Provides interoperability of ePortfolios among platforms.
IMS Question and Test Interoperability	Jan, 2005	Provides proposed standard XML language for describing questions and tests.
IMS Learner Information Package	Jan, 2005	Provides interoperability of user profiles among platforms.
IMS Content Packaging	Nov, 2004	Describes data structures that can be used to exchange data between systems.
IMS Resource List Interoperability	Aug, 2004	Details how structured meta-data can be exchanged between systems.
IMS Enterprise Services	Aug, 2004	Defines how systems manage the exchange of information that describes people, groups and memberships within the context of *e-learning*
IMS Shareable State Persistence	Jul, 2004	Enables the storage of and shared access to state information between content objects.
IMS Vocabulary Definition Exchange	Mar, 2004	Defines a grammar for the exchange of value lists.
IMS Simple Sequencing	Mar, 2003	Defines a method so that any learning technology system can sequence discrete learning activities in a consistent way
IMS Learning Design	Feb, 2003	Provides a model for the different elements of a Unit of Learning (see Chapter 3)
IMS Digital Repositories Specification	Jan, 2003	Provide recommendations for the interoperation of the most common repository functions.
IMS Reusable Definition of Competency or Educational Objective	Oct, 2002	Provides a means to create common understandings of competencies.

Table 2.2: Summary of IMS GLC specifications.

2.4.5 Advanced Distributed Learning

The Department of Defence of the United States of America was interested on providing on-demand training for individuals worldwide. In 1997, they created the Advanced Distributed Learning initiative (ADL) [59] with the following specific goals:

- Identify and recommend standards for training software and associated services purchased by Federal agencies and contractors.
- Facilitate and accelerate the development of key technical training standards in industry and in standards-development organisations.
- Establish guidelines on the use of standards and provide a mechanism to assist DoD and other Federal agencies in the large-scale development, implementation, and assessment of interoperable and reusable learning systems.

The main contribution of ADL is the Sharable Content Object Reference Model (SCORM) [60]. SCORM defines how courseware has to be interpreted and delivered to users by learning management systems. SCORM uses specifications defined by other organisms, especially IMS, and organises them so that their integration offers a system where learning can be created, delivered and reused in an effective way.

Based on XML, SCORM uses IMS Content Packaging to store and deliver content. Support for metadata is provided through the use of the IEEE Learning Object Metadata specification and runtime communication with the Learning Management System can be performed using a JavaScript API, which is similar to the one defined in AICC/CMI. The last revision of the reference model is SCORM2004, which includes sequencing capabilities by means of IMS Simple Sequencing.

2.4.6 IEEE Learning Technology Standards Committee

The Institute of Electrical and Electronics Engineers or IEEE is a non-profit organisation that offers information, resources and services to the engineering community. As part of the IEEE, the Learning Technology Standards Committee (LTSC) was created to *develop accredited technical standards, recommended practices, and guides for learning technology.* Currently, the IEEE LTSC has five active working groups:

- Digital Rights Expression Languages
- Computer Managed Instruction
- Learning Object Metadata
- Resource Aggregation Models for Learning
- Competency Data Standards

Among the accredited standards developed by IEEE LTSC, the most relevant is the IEEE Learning Object Metadata (LOM), which defines the attributes required to effectively describe digital and non-digital learning objects. The standard includes a XML binding of the model that enables its use as part of wider scope recommendations such as SCORM or IMS Learning Design. an XML binding of the model that enables its use as part of wider scope recommendations such as SCORM or IMS Learning Design.

2.4.7 Open Knowledge Initiative

As stated in Sections 2.2 and 2.3, current trend in learning systems is the creation of distributed environments where the learning management system is in fact an aggregation of modules provided by different vendors. The Open Knowledge Initiative (OKI) [61] recognised such relevance of distributed services and proposes a framework where the components of software environments communicate with each other, and with other enterprise systems.

The OKI initiative is based on the Service Oriented Architecture (SOA) and provides software interfaces, called Open Service Interface Definitions (OSIDs) that define the logical services offered by Learning Management Systems. Through the use of OSIDs, applications can be constructed independently of any particular service environment, and integrated in several wider-scope systems. OSIDs contracts are implemented using Simple Object Access Protocol and Web Services Description Language.

2.4.8 CEN Workshop on Learning Technologies

Since it's conception in 1999, The CEN Workshop on Learning Technologies (CEN WSLT) has contributed to the effective adoption of learning specifications and standars. The objectives of the Workshop are [62]:

- Encouraging participation in global initiatives in order to ensure that diverse European requirements are properly addressed by those global initiatives.
- Creating specifications, agreements, guidelines or recommendations where appropriate.
- Providing a forum for the development and implementation of requirements-driven Learning Technologies.
- Carefully examining and taking into account the various effects on learning and training technology standards which are due to the diversity of cultural backgrounds and languages that exists within Europe.
- Publicizing the Learning Technologies Workshop's activities and results to Relevant European projects, Technology developers and end users.
- Providing a forum for discussion for European project initiatives.

2.5 Conclusions

With the goal of providing an overview of the history, present and trends, this chapter summarises the state of the art regarding the use of the World Wide Web in education.

In the early years of the Web, it was used to deliver content in distance scenarios, being the content produced by institutions and consumed by learners. However, the easy content production methods enabled by Web 2.0 tools allows learners to start producing content at the time that interaction among peers is encouraged by the technology. These new affordances make possible to introduce constructivist learning models in distance scenarios where the learner's activity is the central element of the course. The student-centred paradigm suggests the adaptation of the learning material according to the learner preferences, and also encourages the adoption of learning platforms where the learner is able to personalise the content and the environment at their very detail. Thus, the idea of PLE is gaining acceptance in the research community.

Standardisation bodies such as ADL or IMS promote the adoption of common formats that provide reusability to learning material and therefore increase the efficiency of content

creation and delivery. The standardisation initiatives specify how to proceed with content packaging and distribution (SCORM, IMS CP), reusable definition of instructional design flows (IMS SS, IMS LD), assessment (IMS QTI) or learners' profile definition (IMS LIP), just to mention few.

The review of the literature presented in this chapter shows that there are two main problems in the use of Web 2.0 to effectively support student-centred courses. First, the lack of standardisation of the process drives to a scenario where learning material can be hardly reused. The consequence is that the cost of course creation is too high to be affordable by institutions. Second, the activities require to be orchestrated in order to be *learning* activities. However, there is no expressive enough method to provide such orchestration of Web 2.0 based activities. The work presented in this dissertation attempts to overcome these limitations with the definition of an orchestration method that extends IMS LD to integrate Web based tools.

Chapter 3

Capturing the structure of a learning experience

> The art of being wise is the art of knowing what to overlook.
>
> William James

3.1 Introduction

Learning is not a mere observation of knowledge. According to modern theories, an effective knowledge acquisition consists in the relationship among learners, knowledge and activities. Knowledge based paradigms then need to be replaced by activity based ones, where all the elements involved in the course (tutors, learners, learning objects, tools, activities, artefacts, etc.) are coordinated in order to effectively produce learning.

The coordination of all course elements, called *orchestration*, has a great design cost that is worth to be reused more than once. The educational modelling languages appear as a way to formalize learning flows with the aim of achieving reusability. A review of the existing modelling languages is given in Section 3.6.1. Among the existing languages, IMS Learning Design (IMS LD) is the framework for the design and deploy of learning courses that has received more attention from the research community.

Due to its various meanings, it is very common to be confused on the use of the term "learning design". It sometimes refers the process of creating and enacting courses according to instructional design models, and some other times refers the specification adopted by the IMS Global Learning Consortium as the framework to create, package and deploy learning courses. For the purpose of this document, the specification will be referred as IMS Learning Design (with capital letters) or IMS LD, while the course creation process will be called learning design.

Taking the constructivist theories as underlying paradigm, IMS LD has a pedagogically neutral design. That is, it can express any activity based learning method. However, it emphasizes the support for collaborative learning processes and the creation of adaptive content. The low level of adoption of IMS LD contrasts with the intensive research focused on the exploration of the limits of the specification. The drawbacks of the specification are also under study, with the result of the proposal of new extensions intended to overcome them and increase the adoption level.

This chapter gives an overview of the current IMS LD related research. First, as an introduction for the topic, Section 3.2 introduces the theories that support activity based learning and instructional design, as the underlying model of modelling languages in general and the IMS LD specification in particular. Then, Section 3.3 describes the technical details of the specification: its goals, conceptual model and the existing supporting software in the course life-cycle. Current research and experiences of use related to IMS LD are summarized in Section 3.4. Finally, the existing proposals to integrate the specification with other standards and/or tools are depicted in Section 3.5, while other initiatives that extend or replace IMS LD are compiled in Section 3.6.

3.2 Learning activities orchestration

The enactment of an educational experience is the result of a complex process that involves several actors. First, the authors of the material choose a pedagogical method and elaborate the proper content. This material is then allocated in a repository so that it can be accessed by course participants. Finally, teachers and learners get involved in the course and they start interacting among themselves and with the course content. This is a costly process, whose elaboration effort is worth to be reused several times. Here appears the need of a framework that promotes reusability of learning courses.

The enactment of a learning process includes the interaction among the material and course participants. The pedagogical method in use dictates how this interaction should occur, ranging from a simple one-to-one interaction to complex models that require a mechanism to orchestrate the activities performed by the different course elements. The success of the course depends on the correct application of the pedagogical model. Therefore, the orchestration method plays a relevant role in the reusability of a given course. This orchestration could be performed by hand (with human intervention) or it could be automated to some extent.

The formalization of a learning process' orchestration, so that the flow can be replicated in a compliant platform, is called a *learning script*. This section is devoted to explore the affordances and constraints of learning scripts, with special emphasis on the case of IMS Learning Design.

3.2.1 Activity Based Learning

As detailed in Section 3.3, the design of the IMS Learning Design specification was oriented towards the replication of activity-based courses. Therefore, the emphasis is on how to sequence the activity descriptions to the different course participants, and how to monitor the development of the activities, rather than emphasizing the content itself. Sush emphasis on activity-based learning comes from the application of the constructivist theories of human learning. This subsection provides an overview of the theoretical foundations that supports activity-based learning.

Constructivist pedagogical theories, generally attributed to Piaget [1], states that the knowledge appears as the result of the confrontation of the individual experiences and his/her previously existing ideas. The important role played by the experience in the knowledge acquisition turns the learner into an active participant of the learning process. According to constructivism, learning is more effective when implies the active participation of the learner.

Based on the Piaget theories, the social constructivism [2] relates the knowledge acquisition process with the learner environment, especially with the interaction with other

individuals. Vigotsky recognizes the active role of the learner but also emphasizes the relevance of his/her social interactions.

In the last decades, constructivist theories of knowledge have impacted the educational field, pushing a shift from the knowledge-based syllabuses to competence oriented ones. Learning does not longer happen just by reading a set of relevant documents: effective learning happens when there is a balance between *content* and *process* [63]. In an activity-based learning model, educational content is composed by the description of the activities that the learners must develop. These descriptions are usually complemented by a set of resources that would help in the development of the activity. Therefore, the teachers' supporting tasks and evaluation put more emphasis in a proper execution of the planned activities, instead of exclusively focusing in a final assessment of the achieved knowledge.

Problem-based learning (PBL) is a teaching method whose foundations rely in the above described theories. The application of PBL in different disciplines has been deeply studied [64]. It is also a matter of intensive research and practice in Engineering Education [65]. In PBL, students are challenged with open-ended problems, whose solution requires the comprehension of the knowledge that is being taught. To increase students' motivation, the presented problems usually simulate real world contexts. The responsibility of the learning process is assumed by the learners, while teachers appear as mere facilitators of learners.

It is common in PBL activities to require collaboration or cooperation with peers to find a solution to the problem. The collaboration is enriched when the presentation of the problem promotes positive interdependence among the students in the team [66]. That is, the students must feel that each person's effort benefit not only himself, but all team members as well. The collaboration among team members contributes to students reflection on their experiences, so they can construct their own learning.

Activity-based learning needs a well organized sequence of activities (aka. a *learning flow*) to produce learning. The student grades usually consider how well they performed during the whole process, while the final assessment results have less weight. According to that, the replication process of such a course require a well structured method to orchestrate the students learning flow and the teacher monitor tools, reducing the emphasis on content material. One of the goals of educational modelling languages is to provide such replication method.

3.2.2 Instructional Design

Instructional Design is the systematic process of translating general principles of learning and instruction into plans for instructional materials and learning [67]. In other words, instructional design establishes a methodology for the creation of courseware and training programs, focusing on the accomplishment of the learning outcomes and with measurable indicators.

The model proposed in [68] can be considered the first model of instructional design. It was developed as an attempt to use General Systems Theory [69] in the development of military training process. The success of the model on its original context motivated the adoption of instructional design techniques in other disciplines like industry and business [70].

The main characteristics of instructional design are [70]:

- **Learner centred**: The planned activity flow is based on the initial knowledge of the learner and consists in a set of activities to be developed by this learner. The teacher figure is sometimes even not required.
- **Goal oriented**: The courseware is designed so that the learner will achieve well defined learning outcomes. All the activities performed by the learners will therefore

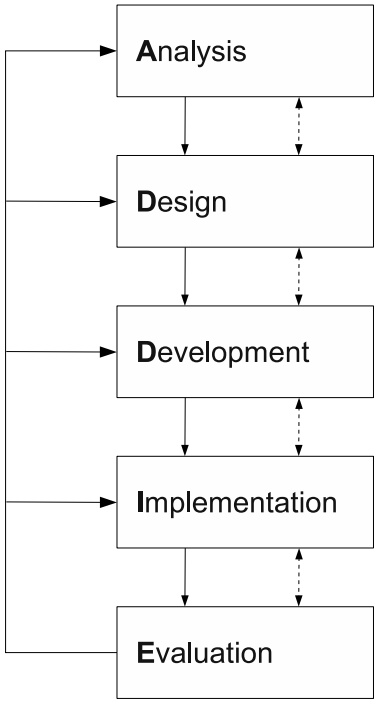


Figure 3.1: The iterative ADDIE model.

be oriented towards the acquisition of a competence, and the course is successful if the competence is acquired.

- **Empirical**: The evaluation of learning outcomes must be measurable in an objective manner. With the same idea, the learner improvements should be mapped into empirical data. Thus, the design of a training program also requires a monitorization method to be provided with the rest of the courseware.

There have been proposed many models for the courseware creation process that complies with the principles of instructional design. It is accepted that all of them share the same basic steps of the ADDIE model: Analysis, Design, Development, Implementation and Evaluation [71]. The first stage, analysis, determines the goals, learners' background and context of the course; next, at the design phase, the pedagogical strategies and their delivery method are defined; the course materials and the required deployment infrastructure are created in the development phase; finally, all the resources are deployed in the implementation phase and the course gets ready to be enacted. As depicted in Figure 3.1, the whole process is affected by the evaluation stage that combines summative and formative evaluation in order to guarantee the quality of the course. The result of the evaluation process usually requires the iterative repetition of the other stages of the model.

Instructional design techniques are used in both online and face-to-face courses. The latter scenario allows the teacher to improvise a new activity in case it is suggested by the circumstances. However, online scenarios lack this flexibility, and thus require a more thoughtful plan of activities. On the other hand, online courses allow storing logs of students' activities in a centralized manner, so that they provide meaningful information for the evaluation phase. Therefore, the analysis and design phases of the course creation are considerably different in both scenarios.

Learning scripts are deeply influenced by instructional design theories, so the creation process of a scripted learning flow usually follows the ADDIE model. In fact, the IMS LD specification can be seen as a framework that facilitates the process of instructional design, where the analysis, design and development phases result in the creation of the Unit of Learning, the implementation is performed in the runtime environment and the evaluation is done on every part of the course life-cycle.

3.2.3 Formalization of Activity Flows

Behavioural scripts are sequences of expected behaviours for a given situation [72]. People do usually follow scripts to accomplish daily tasks. For instance, when someone goes to the cinema he/she usually chooses the film, buy the ticket, optionally buy popcorn, take his place, see the film and then go out. Scripts appear on different parts of our life and they usually are *implicit scripts*. That is, there is not explicit list of actions, but this is the actual sequence that we perform.

Training programs usually make use of *explicit scripts*, so the learner follows a given sequence of activities designed to accomplish a given learning outcome. The mere interaction with the course material, with no established order, does not necessarily produce learning. The construction of the learner's knowledge is more effective when the interaction is guided through a sequence of tasks that has been recognized to be effective for the achievement of the expected learning outcome.

Learning scripts define *who*, *when* and *what* of the activity sequence. Thus, the creation of a learning script is the process studied by the instructional design theories. According to [73], scripts have the following characteristics:

- **Understandability**: The ideal learning script is interpreted by course participants, who perform the activities as they actually understood. A bad self-explained script may result in differences between the ideal script and the actual one. The consequence is a less effective learning process.
- **Flexibility**: Unexpected situations may happen during the enactment of the scripted activities, and there are cases where a disruption of the activity sequence may result in a less effective learning. When the work is in face-to-face sessions, the tutor can improvise new activities to redirect the learning flow. Online delivery of activities reduces the flexibility degree of the activity flow and this fact need to be considered on the course design [74].
- **Degree of coercion**: A script can provide rough guidelines of the activity to perform, or it can detail the activity on its every single aspect. High coercion scripts reduce the gap between ideal and actual scripts, but raise the risk of losing the natural strength of collaborative learning [75].

The formalization of the learning flow into a script implicitly poses a life-cycle of the course, depicted in Figure 3.2. A scripted learning course has three different steps: *authoring*, when the sequence of activities is defined and the course content is created; *deployment*, when course participants are related to the course and the resources are allocated so that they can be accessed by course participants; and *enactment*, when course participants start interacting with course material.

3.3 IMS Learning Design

Initially released in 2003, the IMS Learning Design [6] is considered the *de facto* standard in the field of educational modelling languages [76]. The specification was developed on the

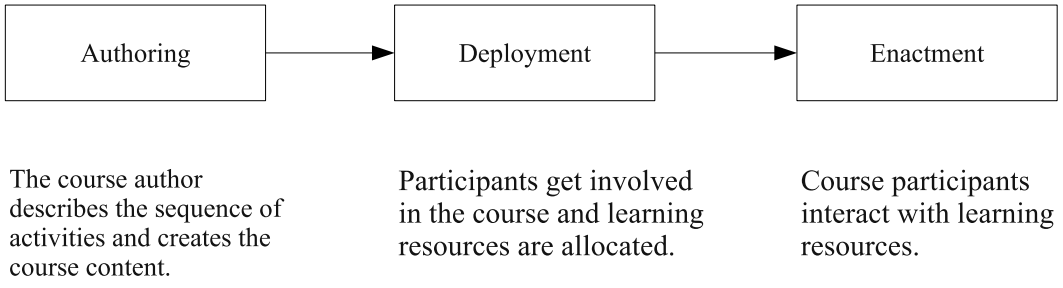


Figure 3.2: Life cycle of learning scripts.

basis of *Educational Modelling Language* (EML), originally designed in the Open University of Netherlands [77]. The IMS Global Learning Consortium supported the model and added compatibility with already existing specifications such as IMS Content Packaging [78] and IMS Learning Resource Metadata Specification [79].

The text of the specification covers the complete course life-cycle: first, proposes a data model that supports the formalization of learning flows; then, offers the XML binding of the data model and defines the exact shape that a course package (called *Unit of Learning*, UoL) must have; finally, details how runtime environment should interpret the different elements of the course. In this section, we summarize the challenges of the specification and its technological description. The section finishes with a compilation of the supporting software currently available in the market and in the literature.

3.3.1 Challenges of the specification

Interoperability is achieved through the use of standards or widely agreed specifications. However, the cost is usually a limit on the possibilities of the technology. The IMS LD specification was designed with the goal of creating a framework that lifts these limitations while maintaining interoperability [80]. Thus, the requirement of IMS LD is to satisfy the following characteristics:

- **Completeness**: The specification must be able to express the teaching-learning process and the required resources: digital and non-digital learning objects; multiple users and roles; online, face-to-face and blended activities and supporting tools.
- **Pedagogical neutrality**: IMS LD is said to support a wide range of pedagogical theories while remains neutral among them. That is, any pedagogical approach can be expressed with the specification, but the terminology of IMS LD is not biased towards any of them and the election of the learning method is a pure decision of the practitioners.
- **Compatibility**: The use of other available standards or specifications must not be limited by the use of IMS LD. For example, learning objects can be still described by LOM [79], and assessment provided by IMS QTI [81].
- **Reusability**: A UoL is created once, but it must be possible to replicate the learning flow several times with a reasonable cost. There should not be any difference in the presentation nor behaviour of all course instances. Thus, UoLs must satisfy the self-containment requisite. That is, all required resources, instructions and materials are included in the course package.

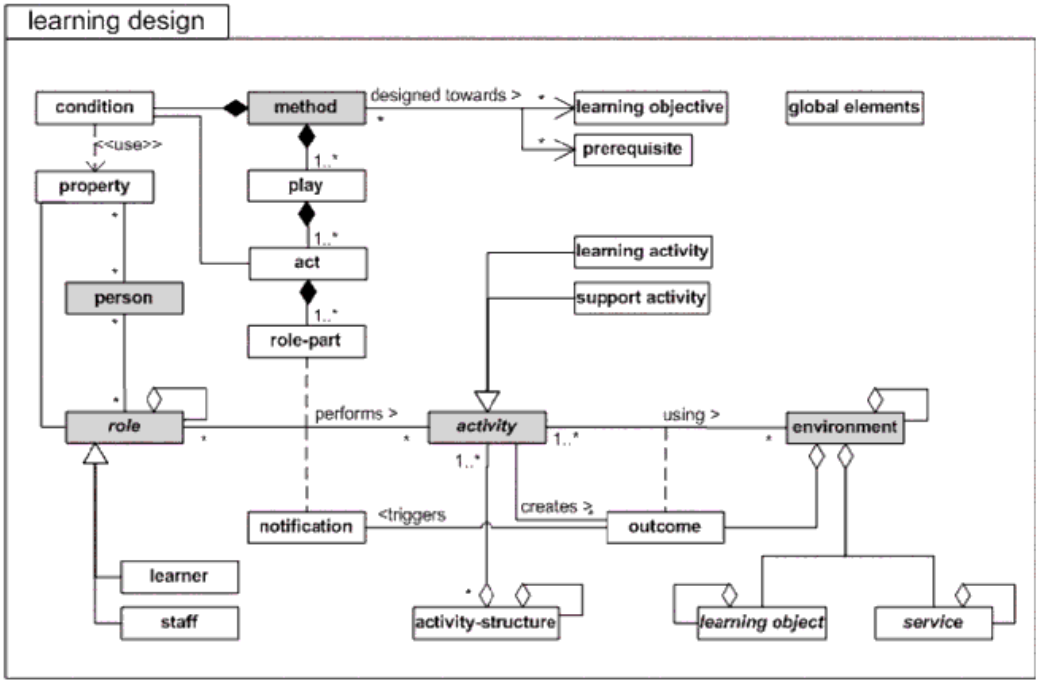


Figure 3.3: Conceptual structure of IMS Learning Design (Extracted from http://www.imsglobal.org/learningdesign/ldv1p0/imsld_infov1p0.html)

The description of IMS LD [6] emphasizes the relevance of the pedagogical neutrality, at the same time that remarks the availability of two possibilities that reinforces this neutrality: collaboration and adaptation.

A course can hold several learning flows, each of them intended for a different course participant. Thus, the course author describes the interactions among the different actors, setting the basis of collaboration. The learning flow can also be designed with different alternatives, to be followed depending on predefined conditions. With this characteristic, it is possible to adapt the activities to the need of the individuals without disrupting the pedagogical method. Collaboration and adaptation features are analysed in Sections 3.4.1 and 3.4.2, respectively.

3.3.2 Technological Description

The IMS Learning Design specification is a framework that defines how authoring, deployment and enactment of learning courses can be performed. It provides a data model that allow IMS LD scripts to express a wide range of pedagogical models and states how deployment platforms should interpret the different elements of the language, and how should they be presented to course participants.

In IMS LD, a course is composed by a *sequence* of activities (aka. the learning flow) that are presented to the end user by their *activity descriptions*. An activity description is a human readable document that explains to course participants the task (online or offline) they are expected to perform. In order to support the learners in the development of their tasks, an *environment* may offer a collection of content material related with the task. An environment is therefore a collection of *resources* and *services* that accompanies one or

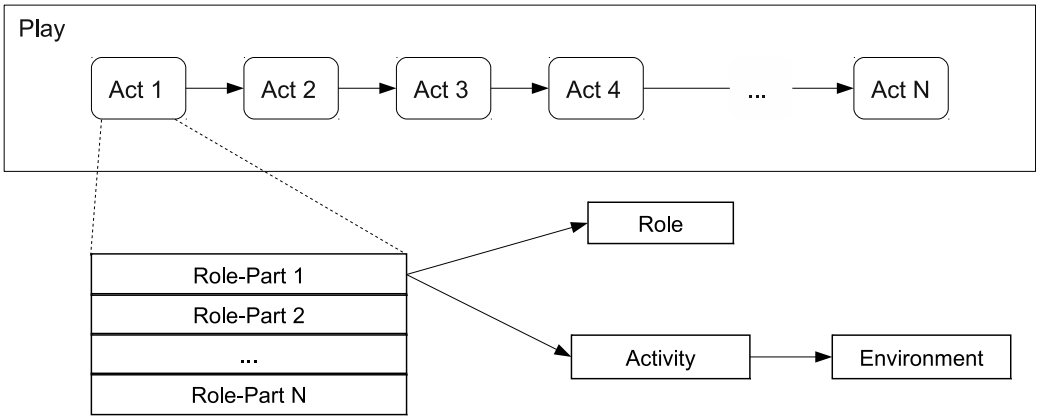


Figure 3.4: Graphical representation of the theatrical metaphor (Extracted from http://www.brookes.ac.uk/research/odl/alt-nl03/Presentations/LD%20presentation.ppt)

more activities of the learning flow. Resources are documents included in the course package, while services are tools whose behaviour should be interpreted by the deployment platform. Course participants are involved in the course by assuming a certain *role* in the learning flow. The role determines the sequence of activities that an individual takes in a course. Typical roles are the teacher and the student, but additional roles can be defined to match the course requirements. A graphical representation of the IMS LD data model is presented in Figure 3.3.

The theatrical metaphor

IMS LD is usually explained with the theatrical metaphor (Figure 3.4). The learning course is then like a theatrical play, where actors perform different activities that depends on the character they are playing. The play is organized in several acts, which are linearly sequenced.

The IMS LD vocabulary is taken from the theatrical metaphor. Thus, a learning flow is a *play*, in which one or more *acts* are delivered in a linear sequence. Inside acts, each participant assumes a *role* and consequently performs a *learning activity*, a *supporting activity* or and *activity structure*. The item that establishes the relationship among activities and roles is called *role-part*. Activities can be related to *environments*, a collection of *learning objects* that support the participants in their tasks. One relevant fact is that the break between acts happens at the same time for all characters and works as a synchronization point.

The theatrical metaphor is valid to introduce the specification to beginners, but there are more supported functionalities that are not part of the metaphor. First, a learning flow is not restricted to include synchronization points: the use of the different structure types allow course designers to create flows where the participants asynchronously reach their objectives [82].

The theatrical metaphor does not allow either introducing adaptive content material: course and participants state can be mapped into concrete values through the use of *properties*. A proper evaluation of these properties, done with *conditions* permits the modification of the learning material depending on the learner's behaviour.

IMS Learning Design Levels

To simplify the understanding and software development, the specification is divided into three incremental levels, described as follows.

First, level A describes the core of the specification. That is, introduces the main elements (role, role-part, act, etc.) and sets the relationships among them, establishing rules for the deployment platform behaviour. In this level, *conference*, *send-mail* and *index* services are defined. A course that strictly follows the theatrical metaphor can be expressed with the elements of level A.

Properties and conditions are introduced in level B. These are the elements that provide support for adaptive content material. Properties are defined by the course author, while property values are set by course participants during the enactment. A change in the value of a property triggers the evaluation of conditions and, if required, the learning flow is consequently modified. The available options to perform this modification are the *show* and *hide* actions, which may affect any structural element (acts, activity descriptions, learning objects, etc.) from level A. The monitor service is introduced in level B and is used to, by means of properties, allow teachers to track students' progress.

The only element introduced in level C is the *notification*. Notifications allow activities to be triggered as the result of a condition evaluation or an activity completion. It is difficult to find in the literature examples of Units of Learning that use the notification element.

Services

As explained above, an environment is a collection of learning resources or services that supports the learners to accomplish their activities. Resources are learning objects defined by course authors and included as part of the course package typically as a document or a link to the media.

On the contrary, services are tools and therefore they are not described by their content, but by their functionality. The specification states how services should behave during enactment, but delegates the final decision to the deployment platform. For instance, the asynchronous conference is typically enacted as a forum, but each platform will use a different forum tool so they will differ in the details.

There are four defined services:

- **Conference**: Can be synchronous (a chat) or asynchronous (a forum). It is intended to put course participants in communication, especially in collaborative learning flows.
- **Send-mail**: With this service, course participants can send a message to one or more of their peers. It is another communication method that complements the conference service.
- **Index-search**: The learners can search the course material to find a specific topic. Depending on the settings, the service can behave as an index of content, an index with links to the content, or as a search engine.
- **Monitor**: Through the use of properties, a course participant can view or modify the property value of other participants. It is originally intended for teachers to track and grade students' progress, but can be use to different purposes.

3.3.3 Supporting software

The availability of supporting software in the market is a measure of a specification health. In the case of IMS LD, software can be produced for two main areas. First, the authoring

stage: it is desirable for practitioners to be able to create UoLs even without knowing the details of the specification. The authoring software should provide an expressive and usable user interface that facilitates the course creation. On the other hand, the deployment and enactment of a UoL requires a compliant platform that imports the course package and react to the different elements as expected by course authors.

An overview of available software for both authoring and enactment is given in [83]. This section summarizes and complements this compilation with the improvements found in the literature since then.

Authoring

Authoring is an aspect in which current tools have not offered a satisfactory solution and therefore hindered the adoption of the specification [84]. This can be the reason why the number of available tools increased in recent years. IMS LD defines a data model and XML binding for it. However, course creators are not expected to create the UoLs by directly editing the raw XML. Instead, authoring tools hide technical details and provide a higher abstract level user interface.

Tree-based editors present the IMS LD data model as a tree structure. The course author can add new activities, learning objects, etc. as new leaves of the tree. One example of this type of editors is *CopperAuthor* [85] which allow the creation of level A designs. *Reload LD Editor* [86] takes the same approach and covers the three levels of the specification. *Recourse* [87] was developed as an evolution of Reload, with improvements on the user interface. Tree-based editors are a representation of the IMS LD data model more user friendly than the XML binding, but still requires understanding of the specification.

To avoid the need of understanding the specification as a requisite for course creation, a different approach is to provide a graphical user interface where activities are represented as boxes and the sequence is represented by connections between boxes. This is the case of *GLM* [88], *Cosmos* [89] and *ASK-LDT* [90].

Some tools, also based on graphical representations of learning flows, were not designed to create IMS LD packages but are capable of exporting the designs in such format. For example, *MOT+LD* [91] exports level A designs, and a comparative between MOT and IMS LD suggests the possibility of upgrading the system to support levels B and C. Another example is *LAMS* [92], a system that integrates authoring and enactment in the same tool. The popularity of this tool is due to its simplicity, but is only able to export level A designs.

Collage [93] follows a pattern based interface that enable practitioners to create collaborative UoLs. With this tool, the course author selects one among several well-known collaborative learning flows (pyramid, think-pair-share, jigsaw, etc.) and populates the activities in the pattern with the proper content. The tool translates the pattern into a IMS LD compliant package. A method based on course planning techniques is followed by *CourseEditor*, presented in [94].

Despite the relatively large number of authoring tools, this field can be said to be immature, since it is difficult to find the trade off between completeness of the specification and usability of the tool.

Deployment and enactment

The result of the authoring process is a zip file called Unit of Learning, which can be loaded into a compliant deployment platform where the learning flow can be enacted. The specification provides guidelines to set the behaviour of deployment platforms, so interoperability can be provided. However, the fine grain details of how the course is presented to the end user depend on the platform in use.

The first tool able to enact UoLs is *CopperCore* [95]. Its implementation is designed as a collection of finite state machines [96] and serves as a reference for other implementation. CopperCore is actually an engine that complies with the specification but lacks of a visualization layer. Complete enactment platforms can use CopperCore as the underlying engine. This is for example the case of the *CopperCore Player*, a proof of concept but a not very user friendly interface.

There exist several tools that use CopperCore as underlying engine. That is the case of *Reload Player*, which is intended to be a previsualization software for the courses created with Reload, but is not expected to be used as a final enactment platform. Another example is SLED [97], which offers a Web interface for managing users and runs, and facilitates the deployment in real scenarios. SLED was the first tool designed to be actually used as a player for end users. The open source nature of CopperCore allows its modifications to incorporate new features. That is the case of *Gridcole* [98], whose added functionality is discussed in Section 3.5.

The first implementation not based on CopperCore, and also the first implementation completely build into a Learning Management System, is *GRAIL* [10]. This software was developed at the Telematics Engineering Department, in the University Carlos III of Madrid, as a contribution for the E-Lane project [99]. GRAIL was created as a module for .LRN [100], a system based on the OpenACS architecture [101]. GRAIL complies with the three levels of the specification and takes advantage of other modules of the LMS to provide the player with some functionalities, as the forum, the file storage or the built in QTI engine. The author of this dissertation participated in the development of the GRAIL player, which is the tool used for the experimental part of the research methodology. Further details about GRAIL are presented in Chapter 4.

Apart from CopperCore, there exists other implementation of the specification [102, 103, 104, 105, 106, 107] that complies with level A, being in some cases the level B support under development. There also exist discussions concerning the integration of IMS LD into Moodle, which is currently the most popular open source LMS in the market [17, 18]. [108] considers the use of Moodle as a level A editor. [109] analyses the mapping among Moodle and IMS LD elements. However, there is no released implementation of built-in player. In general terms, it can be said that the existing players for the IMS LD specifications are in an immature state of development.

3.4 Uses of IMS Learning Design

IMS LD has not reached the adoption level expected for such a model. As analysed in Section 3.3.3, the immature state of the supporting software could be one of the reasons. This low adoption level is deeply analysed in [84]. This article also states that, in spite of the lack of authentic experiences, there are multiple examples of experimental uses of the IMS LD framework. In other words, experiences where the aim is to explore the limits of the specification expressiveness.

IMS LD is being studied in the fields of Computer Supported Collaborative Learning (CSCL) and Adaptive Hypermedia (AH). However, there are more areas where the specification is being applied. This section presents a summary of the most relevant experiences found in the literature.

3.4.1 IMS LD and CSCL

The specification supports the design and enactment of collaborative learning models. The core element for collaboration purposes is the *role* that allows assigning different activities

to course participants. According to [110], the group activity is then defined in the *act*, while the individual tasks are described in the *role-part*. The communication among group members can be mediated through the synchronous or asynchronous *conference service*.

Relevant research analyses the feasibility of applying such models with IMS LD, ranging from pure theoretical discussions to experiences of use and from a wide scope discussion to the support of very specific scenarios. As asserted in [111], IMS LD seems to be able to represent collaborative learning activities, but there is a need of real experiences that prove the assumption.

On the analysis of specific scenarios, [112] uses IMS LD to design courses that require synchronous collaboration among peers. They found that the services defined in the specification are not enough to provide effective synchronous collaboration and propose a new service called *collaboration service*, complemented by a *collaboration activity* and a *group* component as new proposed elements. With a different approach, Collaborative Answer Negotiation Activities (CANA) were described using IMS LD in [113], and the resulting courses were enacted in regular computer science courses.

The application of Problem Based Learning methodologies as IMS LD courses is analysed in [114], where a trial was set up and evaluated. The results shown that the specification was able to design and enact such courses and that teachers recognized the relevance of focusing on the course structure and reusability. However, this was at the cost of much more design time than if the teachers were just focused on content.

Similar conclusions are extracted from the experience described in [115]. This paper, discussed in Chapter 6, details how complex collaborative models can be modelled and delivered on distance scenarios, but usually requires the support of services not described by the specification and with a great cost, in terms of time, used to design and coordinate the course. This high cost causes the design process to be error-prone [116].

The GSIC group, in the University of Valladolid (Spain) has analysed in detail how to best apply collaborative learning with IMS LD (see [117] for an example). Their proposal is based on the use of patterns, which are argued to be an effective approach for course authoring [118, 93]. The pattern based approach is complemented with an extension of the IMS LD model aimed at facilitating the provision of collaborative tools [119], and a platform able to interpret and enact UoLs that include the extension [98]. Also working with patterns, in [120] there is an analysis of how the intrinsic constraints of the *Jigsaw* and the *TAPPS* patterns can be modelled with IMS LD in order to facilitate the administrative tasks. The presented case study revealed that the approach was useful when the number of students was large, or when there are many constraints to control.

Other CSCL related works, not focused on IMS LD, make use of the specification as a complement for the research case. For example, in [121] the analysis of the IMS LD expressiveness for CSCL derives in the proposal of a different language argued to be more appropriated for collaboration purposes. Another example is the work presented in [122, 123] explores a mapping between IMS LD and CIAN, which is a graphical notation for the representation of collaborative scripts.

More recent developments and studies claim that the specification is not expressive enough to cover all the requirements of collaborative learning flows, and try to overcome this deficiency with the proposal of an extension that enriches the expressiveness of IMS LD. This is the case of the work presented in [124] that proposes a large set of new elements to be included in the data model. The proposal given in [125] consist of a way of introducing virtual participants in the enactment of IMS LD courses. They could be used, for example, to complete the number of required participants of a group, or to simulate the enactment for testing purposes.

3.4.2 IMS LD and Adaptive Learning Material

Properties and conditions from level B are introduced in the specification as a mean to create adaptive learning material. A theoretical analysis of the matter is published in [126]. This work states that adaptation is possible with IMS LD, but the specification also poses some drawbacks to consider in the authoring process. First, the required number of conditions grows rapidly with the number of considered learner characteristics. Second, it is not always possible to enforce an order in the activity sequence. Finally, the manifest-centred approach does not provide runtime flexibility. These arguments will be discussed in Section 3.4.4.

Another theoretical, but more specific, review of adaptation capabilities is given in [127], where Adaptive Navigation Support (ANS) techniques are expressed by IMS LD rules. The article works with examples such as *direct guidance* or *adaptive linking annotation*, and concludes that IMS LD is a good candidate to exchange ANS designs. In a similar manner, the synergy between Adaptive Hypermedia (AH) and IMS LD is said to be beneficial for both models in [128], which also recognizes the need of further research on the topic.

A deeper analysis of the application of adaptation with IMS LD is given in [129, 130]. According to this paper, adaptation techniques can be grouped in three main areas: interface based, learning flow based and content based. The rest of techniques can be considered as a combination of the basic three. The specification states nothing about interface based adaptation, so it will be supported as long as the IMS LD players provide the required functionality. Learning flow based and content based are supported by means of the *show-hide* action, that can be applied to learning activities or to particular parts of XHTML documents appropriately tagged by a *div* element. The UoLs that support this study are available at the Learning Network for Learning Design at The Open University of the Netherlands [131].

An approach more based on case studies rather than on a theoretical analysis is given in [132]. This document takes data from three already existing courses being imparted in Moodle and compiles all the modifications that tutors performed during the runtime phase of the courses. The aim of the study is to evaluate whether these real case modifications could have been performed if equivalent IMS LD courses were applied. The model proved to be expressive enough to describe the wide range of adaptations performed.

Other researches also use IMS LD in the context of adaptive content. For example, the aLFanet platform presented in [133] is a IMS standard based system for the creation and delivery of adaptive material that combines the use of IMS LD, IMS QTI and IMS LIP. Another example is the work presented in [134, 135] aims at the authoring stage of adaptive content, exploring the possibility of automatic generation of adaptive UoLs. The inputs to the system are intrinsic characteristics of users and the desired and achieved competences in the learning process. Finally, the work presented in [136] states that IMS LD expressiveness is not enough for context aware adaptive purposes, and proposes an extension which is in an early development stage.

3.4.3 Other users of the specification

As stated in the best practice and implementation guide [137], IMS LD was developed with the need of "reflecting learning experiences that are collaborative or group-based", but also to "describe and implement different kinds of learning approaches". That is, despite the emphasis that the specification gives to adaptive content and collaborative models, there are other learning approaches whose support in IMS LD is being explored in recent research. A compilation of these efforts is presented as follows:

- The use of mobile devices to deliver learning material, providing context and device awareness is discussed in [138]. The presented prototype does not clarify the role of

the specification in the system, but means a step towards the use of IMS LD in mobile devices

- In [139], IMS LD is used to create a course flow that replicates the Scrum programming methodology [140], in order to be used as a teaching method in a higher institution. The prototype uses CopperCore with Facebook as the presentation layer.
- The work published in [141, 142] is focused on the study of the relationship among ontologies and IMS LD. They propose an ontology aimed at overcoming the expressiveness limitations of the IMS LD XML Schema and discuss its application to a particular case study.
- How to promote reuse of UoLs is analysed in [143]. The authors defend IMS LD as a suitable method for the recognition of re-usable course patterns. When a repository is well populated it can be possible to capture more successful learning designs that can be used as templates.
- The UML4LD experimentation [144] is an attempt for the automatic generation of UML diagrams from courses described with IMS LD.
- The relevance of the terms *competence*, *evaluation*, *artefact* and *feedback* is considered in [145], where a modification of the specification including those concepts in the data model is proposed with the goal of improving assessment support.

There exists more IMS LD related literature, with an increasing number of publications in recent years. Some of the existing works may not present relevant contributions to the existing knowledge, but reveals a rising interest on the specification and its application on real scenarios.

3.4.4 Problems of the specification

The intensive activity related to IMS LD reveals that it was received with expectation, and that there are many efforts trying to push the adoption for the specification in authentic situations. It is even considered the *de facto* standard in instructional design [146, 76, 147, 106, 148]. However, IMS LD has not reached a widespread adoption, as it was expected few years ago. The specification presents several advantages and innovations when compared to other systems, but the original design also present some drawbacks that limit its usage. Here we summarize the main problems reported in the literature.

A comprehensive report on IMS LD problems is presented in [84], where the identified problems are:

Adoption. The steep learning curve that IMS LD poses to practitioners makes them sceptical when considering the benefits of the specification. The existing tension between functionality and complexity do not contribute to relax this learning curve. Furthermore, the adoption by institutions would require a difficult organizational change. As a result, training programs have not success on promoting the use of IMS LD

Life Cycle. There is no defined method to incorporate changes on live courses. Edition features are not integrated in most current systems[1] and there is an incomplete cycle between the authoring phase, the deployment phase, and the enactment phase and then again with the authoring phase. In summary, the specification is manifest-centred [126] and the consequence is the lack of flexibility during runtime.

[1] GRAIL is the only current player that incorporates edition features

Level B Notation. A good enough usability for authoring level B elements has not been reached in current editors. First, the assignment of property values are performed from content (using the *set-property-value* statement) and authoring software do not provide a method for synchronization between the content and the manifest. Second, conditions are expressed in a declarative manner and it may be counter-intuitive to express flows with such vocabulary. Finally, the number of required conditions becomes too large when the number of considered student characteristics increases. That is, level B is powerful but seems to be too complex to be used without patterns, and patterns are not straightforward to be developed.

Interoperability. It is not defined how to exchange data with other tools, such as assessment ones. As a result, there is a problem with data flow [116]. There is also an agreement on the need of more services to be proposed, as well as a mean to bidirectionally exchange data with these services. The need of new services is the main topic in this dissertation, so this problem will be discussed in Section 3.5.

Usability & Utility. There are some conflicting facts that hinder a good usability of IMS LD tools. First, the activity based model in which the specification is based does not always match with teachers' conception of a course. Second, the visualization during authoring is not the visualization during enactment. In fact, each player can provide a different interface, so the course author cannot know how the course will be presented. Finally, it is not clearly defined how the learning path should be presented and there is not trivial answer to this question.

The commented problems concern the specification as a whole, in the sense that they are relevant regardless the pedagogical approach to be applied. Other critiques are more focused on the lack of expressiveness for a particular learning method. For example, [124] states that collaboration tools are not enough, while [145] claims for better support for assessment. These field-specific critiques are usually the seed of extension proposals, and will be discussed in Section 3.6.2

This PhD thesis focuses on the interoperability problem. More specifically, the presented proposal is a framework that enables the integration of third party tools in IMS LD courses, with the support of bidirectional exchange of information with these tools.

3.5 Service integration in IMS LD

As stated previously, one of the documented drawbacks of the specification is the impossibility to integrate arbitrary tools into Units of Learning, including the exchange of information with the tool during the runtime. Currently, tools are integrated as services, but the only defined tools are send-mail, conference and index-search. Furthermore, there is not defined method to gather information from these tools. It is not allowed, for example, to adapt the learning flow depending on how many contributions are in a forum.

The lack of integration is a well-known problem, and there exists several initiatives oriented towards the use of functionality from external tools in the context of IMS LD courses. This section is devoted to summarize the existing initiatives.

3.5.1 Interoperability with other standards

The IMS Global Learning Consortium provides several specifications in for very different purposes in the context of *e-learning*. It is stated, for example, how to achieve interoperability in assessment [81] and how to model user profiles [149]. It is therefore relevant to provide a method to interconnect all these specifications so they can work together.

IMS Question and Test Interoperability

The IMS consortium defines the creation and behaviour of *e-learning* assessment in the so called IMS Question and Test Interoperability (QTI) specification. In the same way that IMS LD does for learning flows, QTI provides a data model and a XML binding. It is also specified how platforms should interpret QTI elements and what information can be stored.

Since assessment is not provided as part of IMS LD, the IMS consortium offers a way to integrate IMS QTI questionnaires into Units of Learning, as specified in the IMS Question and Test Interoperability Integration Guide [150]. Then, if the IMS LD player detects the inclusion of a QTI resource, appropriately tagged as such, the execution of the item is delegated to a pre-configured IMS QTI player. The method also allows properties in the two systems to be synchronized.

These integration guidelines are provided for the specific case of assessment when it is delivered through IMS QTI. Therefore, this method cannot be used to integrate any tool into a Unit of Learning. Furthermore, as explored in [151], the method does not provide complete support for emerging forms of assessment.

SCORM and IMS Learning Design

SCORM is the recommendation of Advanced Distributed Learning (ADL) to package and deploy *e-learning* courses. The SCORM2004 version supports adaptation through the use of IMS Simple Sequencing, though this feature is rarely used. The joint use of SCORM2004 and IMS LD is discussed in [152]. According to this paper, there are three levels of integration:

- *Minimal integration* just requires the URL of the SCORM course to be included as a learning object in the Unit of Learning. The method is quite simple, but the URL is not always available when the UoL is being created. Furthermore, the approach does not allow the IMS LD and SCORM courses to exchange information among them.
- *Packaged integration* requires the SCORM zip package to be included as a resource properly tagged in the IMS LD package. When the IMS LD player finds the SCORM resource, its execution is delegated to a compliant platform. This method, supported in the GRAIL player, does not still allow information exchange among courses.
- *Run-time integration* includes the synchronization of property values among the two systems, allowing IMS LD courses to be adapted depending on values generated in the SCORM object. This approach is prototyped and tested in [153].

SCORM integration with IMS LD shares some elements with the above described QTI integration, and also shares the same drawbacks. In the authoring side, there are differences on how SCORM Sharable Content Objects (SCOs) are incorporated into a Learning Design: whereas QTI content is essentially an XML file which needs to be processed and rendered, a SCO is typically a single HTML page which may contain JavaScript. This page needs to have access to an API-Adapter object within the page (or frameset hierarchy) as it is being used by the engine.

IMS Tools Interoperability

Recognizing the possibilities offered by third-party tools in the context of *e-learning* courses, the IMS consortium released in 2006 the IMS Tools Interoperability Guidelines (IMS TI) [154]. The goal of this specification is to allow non-LMS Web based tools to be used in a LMS thanks to the use of a Web Service infrastructure in both the LMS and the external tool. As an evolution of the same idea, the IMS Basic Learning Tools Interoperability

specification was published in 2010, while the IMS consortium is actively working on IMS Learning Tools Interoperability, a more complete version of the idea.

These three specifications are oriented towards "a formal, negotiated deployment process" [154] between the third-party tool (the tool provider) and the LMS (the tool consumer). Thus, IMS imposes a model to be fulfilled by both provider and consumer so it restricts the tools that can be integrated to those that comply with the specification.

IMS TI, the first of these three proposals, explicitly excludes the integration or "Multi-learner Tools, workflow enabled Tools", which is a clear restriction to use the model in the context of IMS Learning Design. The more recent proposals mention the concept of role but, in the case of workflow enabled tools, do not discuss the impact of the integration in the course life-cycle which is essential for a complete integration.

3.5.2 Non standard-based approaches

The joint usage of the capacity of IMS LD to orchestrate activities with the use of arbitrary tools is a matter of research whose result is the existence of several initiatives, summarized as follows.

The CopperCore Service Integration Layer

The interoperability problem was envisaged by CopperCore developers, who proposed a Web service based architecture that enables the communication of the IMS LD engine with other tools through the so called *CopperCore Service Integration Layer* (CCSI) [155]. This layer allows new services to be added and extend the Learning Design Framework. These services could be either resources that a UoL may want to access at run-time (e.g. a chat service) or alternatively, the service could be based on a different *e-learning* specification. For example, the already mentioned QTI integration was performed through Interoperable Segments (APIS) [156] using CCSI.

The CCSI layer is intended to be used during the enactment phase of a course. That is, allows the communication among services but do not specify how services are defined by course authors and how services are instantiated. Without a recommendation or a best practices guide, the integration process requires IMS LD expertise and it is said to be time consuming, which prevents for an agile integration of arbitrary tools.

e-Adventure

The Unit of Learning presented in [157] is an interesting application of the above-mentioned CCSI architecture. The core element of the presented learning flow is a point-and-click adventure game developed for the e-Adventure engine [158]. Thus, the IMS LD player and the e-Adventure engine are integrated so that the story of the game depends on the user profile, and the course property values are assigned by the game engine.

The prototype reaches bidirectional exchange of information during enactment and demonstrates the capabilities of CCSI. However, the intrinsic problems of the architecture are still present and the cost of the UoL creation is still too high.

Gridcole

One of the earlier initiatives in the field of integration is the use of grid tools through *Gridcole* [98]. The proposal contemplates the whole course life-cycle: the authoring stage uses the *groupservice* extension (see Section 3.6.2) in order to describe the tool to use during the enactment. Gridcole is the name received by the CopperCore modification that is able to interpret the *groupservice* and to deploy the corresponding tool.

The development of Gridcole is oriented towards the use of grid based tools. This fact restricts the integrable tools to those that comply with the *Open Grid Services Infrastructure* model and provide non persistent grid services. Another restriction of the proposal is the lack of support for adaptive content. In other words, property values cannot be set according to the information handled in the grid tool.

Wookie

The work developed at the University of Bolton and presented in [159, 160] promotes the use of widgets as a mean to provide a rich set of services while maintaining portability across different IMS LD platforms. The proposal includes the use of *Wookie*, a server for the easy creation and deployment of widget based tools. The widget server was developed with the needs of IMS LD in mind, but it has been created as an independent server, and integration has already been demonstrated in Elgg and Moodle.

According to the Wookie proposal, widgets included in the course flow can be defined with the existing elements in the current version of IMS LD. The method to indicate the presence of a widget service is the use of specific parameters of the already existing conference service. Then, the IMS LD player interprets the service as a widget, and allows the proper widget to be selected.

The advantage of this late-binding approach is the possibility of using the current specification with the need of neither extensions nor modifications. However, a more verbose description of the required functionality would be required in order to enhance reusability of the model. In other words, how do course managers know the widget they have to choose and how to configure it? A poor widget selection, caused by the not verbose description of the requisites, may result on a non successful learning flow. Besides, the proposal does not include any method to synchronize properties among tools and the creation of adaptive material is therefore not supported.

Rest-based approaches

The use of Representational State Transfer (ReST) architectural style to integrate third-party services is explored in [161]. A simple methodology for the integration of ReST based services is provided, with the argue that new services can be integrated with a reasonably cost. The proposal includes, as a proof of concept, the integration of the wiki functionality via three different wiki tools in the market. In this work, service-specific primitive operations have been considered and further work is required to cover the complete life-cycle of the service, from authoring and enactment to monitoring to interoperability facilities

Generic Service Integration

This Thesis proposes the *Generic Service Integration* framework (GSI), a complement to IMS LD that provides support for the integration of third-party services. The proposal covers the complete course life-cycle (authoring, deployment, enactment) and allows for the bidirectional exchange of information among IMS LD and the third party tools. The complete description of GSI is given at chapter 5.

3.6 Other approaches

Modelling languages can be designed to be case-specific, or they can follow a neutral approach. The former are only suitable for a short number of scenarios, while the latter take the risk of being a *one-size-fits-all* alternative where fine grain details of specific fields cannot

be modelled. In the field of instructional design, IMS Learning Design is the educational modelling language that has received more attention from researchers. However, its expressiveness is not always considered enough for certain purposes where alternatives to the specification are required.

On the one hand, the study of IMS LD drawbacks has pushed the proposal of several modifications of the specification. These extensions are usually focused to improve the support of a given methodology, such as the case of collaborative support. On the other hand, there are many more modelling languages whose functionality can be compared to IMS LD. This section presents a compilation of both alternatives: extension to IMS LD and other modelling languages and/or instructional design systems.

3.6.1 Other modelling languages

The formalization of learning flows into modelling languages is a recent matter of research. One of the earlier works is *Inscript* [162], a scripting language developed at the Dutch Open University to support the courseware production process in the design stage. The script concept evolved into a more mature language, called *Educational Modelling Language* (EML) [77], which was chosen by the IMS consortium as the basis of the IMS Learning Design framework. There are more modelling languages for instructional design, [163] provides a classification method to select the more appropriate language on each case. How to design languages into the practice of instructional design is discussed at [164].

IMS Simple Sequencing [165] is another method to describe learning flows. One of the advantages of this approach is that it is included as part of the SCORM2004 recommendation [60]. It also allows the creation of conditional flows. However, Simple Sequencing is intended to be used by a single learner so it does not allow the creation of collaborative activity flows.

One of the most popular systems for the creation and enactment of learning flows is the *Learning Activity Management System* LAMS [92]. Strictly, LAMS is not a modelling language, but a platform to graphically create learning designs, with the possibility of also enacting them. The platform was based on the conceptual model of IMS Learning Design, but it is not manifest-centred. This fact provides more flexibility due to designs can be modified after at the runtime. LAMS is focused on a visual an intuitive authoring process of learning designs (which is the main cause of its success) and the possibility of real-time tracking of students. Due to the similarity of the proposals, LAMS models can be exported as IMS LD level A compliant packages, but conditional delivery of activities is not supported.

The analysis of IMS Learning Design's complexity caused the creation of *Simple Learning Design 2.0*. In [166] it is argued that the model proposed by IMS is too complex to be actually adopted and that the surrounding circumstances (lack of IMS support, publication of Common Cartdrige) hinders the success of IMS LD. Due to these reasons, a new specification is proposed in [167] that, according to the authors, is simpler than IMS LD and have the required functionality to be adopted as a future standard.

A research for the simplification of IMS LD is also the case of the work presented in [168]. The authors argue that a separation of the modelling problem in several parts is required. Their proposal performs the division and analyses the relationships among parts, which in this context are called *perspectives*. Finally, they consider how IMS LD elements are reflected in each of the twelve defined perspectives. As a result of their work, they propose the *Perspective-oriented Educational Modelling Language* (PoEML) [169] as simpler alternative to IMS LD. Support for PoEML is currently under development [170].

A different view of the course process is presented in [171]. This paper states that the principal problem of IMS LD is that a learning course is considered as a mere sequence of activities. From the point of view of this article, a pedagogical activity is a set of exchanges

	Description	Field of Use	Supporting Software
Simple Sequencing	Describes conditional activity sequences	Neutral	SCORM2004 compliant
LAMS	A platform, not a language. Describes linear activity sequences	Neutral	LAMS platform
Simple LD 2.0	Simplification of IMS LD	Neutral	Not yet available
PoEML	Authoring units of learning focusing on 12 defined perspectives	Neutral	Not yet available
LDL	Learning situations are exchanges among participants. Focused on collaboration and adaptation	Neutral	LDI
CPM	UML based modelling language focused on problem-based learning	PBL	CASE

Table 3.1: Comparative summary of educational modelling languages.

occurring among the activity participants, which is intrinsically evolutionary and unpredictable. These are the basis of *Learning Design Language* (LDL), a language for course creation that emphasizes the support for collaboration, activity observation and activity adaptation. The *Learning Design Infrastructure* is the system that supports the execution of LDL situations.

Focused on modelling problem-based learning activities, the *Cooperative Problem-Based Learning Metamodel* (CPM) is presented in [172]. CPM defines 35 concepts; most of them are similar to IMS LD concepts (Learning Phase, Activity, Activity Structure, Role, Resource, Objective, Post-Condition, etc.); other ones do not exist in other EMLs because of their relation to Problem Based Learning (Subject, Task, Obstacle, Pbl Constraint, Success-Criterion, etc.). The proposal, aimed at the authoring phase, defines elements that reuse and extend UML features. The result is a UML-based method for the creation of learning activities. Authoring of CPM designs is supported by the CASE tool. The execution of a CPM model requires its transformation into an interoperable format such as IMS LD, while there are not CPM enactment platforms.

3.6.2 Modifications of the specification

In the field of collaborative learning, a new type of service is proposed in [119]. The new service, called *groupservice* generically describes tools that support collaborative process that may happen in the learning flow. That is, the service could be used to describe different tools, all of the within the range of collaborative models. *Gridcole* [98] is the system that supports the proposed service type. It is a modification of CopperCore, and is oriented towards the support of grid tools.

The extension via new services is the proposal of [112]. This work, in agreement with other already referenced articles, considers that existing services are not enough for collabo-

	Description	Field of Use	Supporting Software
groupservice	A service type that can model tools for CSCL	Collaborative Learning	Gridcole
Collaboration Tool Service	Service that provides static-info, dynamic-info, discussion and voting	Collaborative Learning	Beehive
Extension for group formation	New element for dynamic group management, and modifications on existing elements to support the new one	Collaborative Learning	Not yet available
Extension to express assessment	New elements that model artefacts, evaluation, marking and feedback	Assessment	Not yet available
Improvements in adaptation	Support for context awareness	Adaptive content	Not yet available
Inclusion of ITS	Extension for the inclusion of virtual roles in courses	Collaborative Learning	Not yet available

Table 3.2: Comparative summary of IMS LD modifications.

ration purposes, and defines a new service called *Collaboration Tool Service*. The proposed service consists of four core elements: Static-Info, Dynamic-Info, Discussion, and Vote, which are argued to be able to describe all tasks needed to assemble any synchronous collaborative learning activities. Other two elements, *collaboration activity* and *group* are proposed. However, the overlapping of these elements with the already existing ones is not discussed. The extension is applied in a software called Beehive, build as a module of the .LRN system.

The work presented in [124] critiques IMS LD from five different perspectives: group creation, properties, services, activities and sequencing. In fact, the discussion claims for the need of a dynamic group management and the other critiques are consequence of the first one (for example, the need of group-scoped properties appears when groups are introduced). The proposed solution is the inclusion of new elements in the IMS LD data model, and the extension of the existing ones. The end goal of the proposed modification is a better support for collaborative activities where flexible groups are required in order to success in the enactment of the learning flow. The proposal is in a very early stage of development and there is no existing prototype that supports the model.

Another extension of the datamodel is proposed in [173]. In this case, the document argues that assessment support do not cover the minimal requisites, and introduces new elements that model the artefacts, evaluation, marking and feedback. The prevailing idea is to improve the description of learning designs without introducing too much complexity in the framework. The authors consider the modification of CopperCore as the more realistic approach of supporting software development, and plain this task a future work.

There are other works [136, 125] that claims for the need of extensions in the fields of adaptive learning and integration with intelligent tutoring systems. They are in an early stage of development and reveals that the interest in IMS LD is still alive. Table 3.2 summarizes the IMS LD modifications presented here.

As stated in [166], one of the problems of IMS LD is the absence of dialogue between the IMS and the LD community after the publication of LD in 2003. Consequently, there was no

real discussion between the IMS working group and the LD community, so the proposals for the extension and modification of LD coming from researchers, teachers and instructional designers have been met with silence.

The work presented in this dissertation can be considered as an extension of the specification that provides a framework to integrate third-party tools in learning designs.

3.7 Conclusion

The creation of effective units of learning is a complex task that has a great cost. First, the learning scenario has to be carefully analysed in order to select a proper pedagogical method. Then, the activities are designed and the content (usually multimedia) is created. Finally, the resources involved in the course are allocated and the actors start the learning process. Effective methodologies that minimize the cost of the overall process are under research in the field of instructional design.

In order to maximize the effectiveness - in terms of cost - of the process, there is a need to reuse successful learning courses more than once. This task is not straightforward due to the complexity of the needed orchestration among resources and participants. This is the reason why courses are formalized in a computer interpretable manner. Thus, the replication of courses requires their formalization in existing modelling languages for their latter replication on platforms capable to interpret these languages.

The study and creation of educational modelling languages is a relatively recent matter of research. There are several initiatives aimed at the creation of such models. Among them, the framework that has received more attention from the community is IMS Learning Design. Its design was based on the previously existing EML and it was adopted by the IMS Global Learning Consortium as the reference model.

The IMS LD specification is build upon a constructivist approach: it is focused on activities, rather on content. A Unit of Learning (a course described with IMS LD) is a sequence of activities whose completion may require the support of learning objects and/or peers. The specification is said to support collaborative learning and adaptive content, among other pedagogical approaches.

Despite the wide research surrounding IMS LD, the model has not been yet adopted by educational institutions. It is said that the conceptual model is too complicated for non-technical practitioners and that a more high level description of courses is required. In such way, supporting software tries to hide the conceptual model behind graphical interfaces, but the tools are not yet mature enough. Furthermore, the specification has some other drawbacks that hinder its adoption.

Two of these drawbacks documented in the literature are the lack of flexibility during the enactment of learning courses and the lack of integration with other tools, especially when an exchange of information is required. Some extensions and/or modifications have been proposed in order to overcome the commented problems. In the particular case of tool integration, none of the existing proposals offers a solution that covers the complete course life-cycle and supports the integration of generic services with a bidirectional exchange of information among IMS LD and the services.

This dissertation contributes with a proposal for the solution of these two limitations. First, the study of flexibility in scripted courses result in the implementation of new features in GRAIL that allow instructors to react to unexpected situations during the course enactment. The second contribution of this disseration is the proposal of a framework that allow the integration of Web 2.0 tools in the context of IMS LD courses, including bidirectional exchanges of information during enactment. The proposal, comprehensively detailed in Chapter 5, affects the whole course life-cycle: authoring, deployment and enactment.

Chapter 4

Support for IMS LD in .LRN

> An expert is a person who has made all the mistakes that can be made in a very narrow field.
>
> Niels Bohr

4.1 Introduction

As shown in Chapter 3, the data model provided by IMS Learning Design enables its use as a pure modelling language for educational courses. That is, practitioners can use the IMS LD vocabulary to logically structure the learning material. However, the objective of IMS LD is not limited to act as a modelling language. The specification states how the structural elements must be enacted. The result is the existence of compliant players (also referred as runtime environments) capable to orchestrate the course resources attending to the instructions of the course model.

The IMS LD specification states how the elements should be understood and interpreted by the runtime environments. However, the guidelines are flexible enough so that the different players can comply with the specification at the time they provide particular behavioural solutions for the open problems. These particular features may influence the course enactment with the provision of extra resources that can enrich the learning activities. The development of a IMS LD player is not merely the implementation of a workflow engine and should be focus on the improvement of the learning experience.

This chapter describes the implementation of a IMS LD player in .LRN, called GRAIL (Gradient-lab RTE for Adaptive IMS LD in .LRN) [10]. The proposals of this dissertation have been implemented in GRAIL, which has also served for the experimental phase of the methodology. The description of this software includes the decisions that derived in the implementation of the player, the technical description of the tool and the main characteristics of the tool from the end user's point of view. This chapter also presents the functionality implemented in GRAIL oriented towards the support of enactment flexibility, which is recognised as one of the limitations of the specification.

The rest of this chapter is organised as follows. The decisions of the design and development process and the underlying technology of the GRAIL tool are presented in Section 4.2. The description of the tool details the steps taken on the different phases of the course life-cycle when a UoL is enacted in GRAIL and the integration of such life-cycle in the .LRN

platform. Then, Section 4.3 introduces the concept of flexibility in learning courses and details the solutions proposed in GRAIL.

4.2 Runtime environment implementation

GRAIL (Gradient-lab RTE for Adaptive IMS LD in .LRN) is the IMS-LD run-time environment implemented in .LRN [100]. It has been conceived to be used within the context of a .LRN community, a set of users sharing resources such as documents, forums, calendar, schedule, etc. GRAIL is the first player integrated in a Learning Management System. The integration provides the oportunity to use the tools of the LMS in the contex of IMS LD courses. The implementation of the specification requirements translates into a set of decisions that need to be taken by the design team. They relate to important aspects of the usability and effectiveness of the run-time environment and therefore need to be carefully considered. It follows a brief description of the aspects taken into account when designing GRAIL.

4.2.1 Relevant decisions

Although they are not part of this dissertation, some developments and decisions held by the Gradient Laboratory[1] influenced the work presented in this document. Among others, the most straightforward influence is the use of the .LRN platform as the basis of the software development. This subsection draws a picture of the reasons that leaded to the proposals presented in this dissertation.

The adoption of .LRN was decided in the context of the E-LANE project [99], an initiative funded by the European Commission through the @LIS programme. The goal of the project was to reinforce the partnership between the European Union and Latin America in the field of the Information Society, focusing on the case of e-Education [174]. The three objectives of the project were:

- **Software development**: The creation and/or integration of solid open source *e-learning* applications within an existing LMS. Sustainability after the conclusion of the project was a requisite and it was promoted by the use of open source software and open *e-learning* specifications.

- **Content production**: As important as the functionality is the content available at the platform. Several courses were produced within the context of the E-LANE project, and deployed in the chosen LMS.

- **Methodology**: The content itself does not promote sustainability, which is better achieved through an open, well-defined methodology to develop new content. The proposed process used *DocBook* [175] for content development and SCORM for delivery and deployment of the materials [176, 177].

At the beginning of the E-LANE project, more than 50 open source LMS platforms existed in the market. Among them, the OpenACS Web toolkit and its community-based approach with .LRN offered a scalable, highly modular system that allowed quick and solid development of new features. Furthermore, the existing community of developers, supporters and end users guaranteed the sustainability of the initiative. More details about the platform can be found in Section 4.2.2.

[1] at Telematics Engineering Department at the University Carlos III of Madrid

The methodology for content creation, which included the use of DocBook and SCORM, was chosen because its feasibility with already existing tools. This decision emphasised the sustainability of the project. However, the absence of sophisticated (collaborative, adaptive) sequencing tools was identified as one of the flaws of the model. This was the initial seed that resulted in the implementation of IMS LD support in .LRN.

Once identified the need of IMS LD support in .LRN, the possibility of integration with CopperCore was evaluated. This integration would have offered a solid player and a quick result at the cost of the duplication of data in the databases of both systems (.LRN and CopperCore). On the contrary, the implementation from scratch offered the opportunity to reuse lot of existing functionalities in the LMS and the possibility of future improvements of the tools at a reduced cost. The latter option was better considered and therefore the GRAIL tool (see Section 4.2.3) was developed. The expertise acquired with the .LRN use during the E-LANE project and the modularity of the underlying architecture allowed for the creation of new functionalities, which was one of the factors that motivated the developments performed as part of this dissertation.

4.2.2 The .LRN platform

.LRN [100] (pronounced as *dot-learn*) is an enterprise-class open source platform for supporting *e-learning* and digital communities. The tool was originally developed at the Massachusetts Institute of Technology as a virtual learning environment, and it then evolved into a comprehensive platform including not only *e-learning* support but also generic Web resources.

Technical description

The platform is based in the Open Architecture Community System (OACS) [101], a toolkit for building scalable, community-oriented Web applications. The toolkit structure is highly modular and .LRN is a set of modules that provide the additional features to deploy an *e-learning* environment.

Both .LRN and OACS are tightly integrated with a relational database; PostgreSQL and Oracle are currently supported. The Web server that handles requests at the basic level is AOLServer [178], the America Online's open source Web server. One of its features is the integration in its core of a multi-threaded TCL interpreter which provides an effective solution for industrial strength type of services such as those present in large higher educational institutions.

Figure 4.1 shows the platform architecture, while Figure 4.2 presents a simplified and more understandable diagram. The OpenACS Web toolkit allows for easy development of new applications (called modules, in this context) and .LRN is one of the already existing modules. The .LRN module is accessed as an environment oriented towards the management of communities. Thanks to a well defined plugin policy, existing OpenACS tools can be used in the context of a community if the corresponding plugin has been implemented. Due to the simplicity of plugin's development, most of OpenACS tools are commonly used through the .LRN environment. Actually, some of the existing tools were initially conceived to be used in the .LRN context and only make sense if they are used inside a community.

Design principles and functionality

The modular structure of OpenACS allows for very fast customisation and prototyping of new applications. The user space is organised through a customisable set of portlets, each of them offering access to one of the various available services. The underlying toolkit provides a set of Web functionality most of which is suitable to be adopted by the *e-learning* platform.

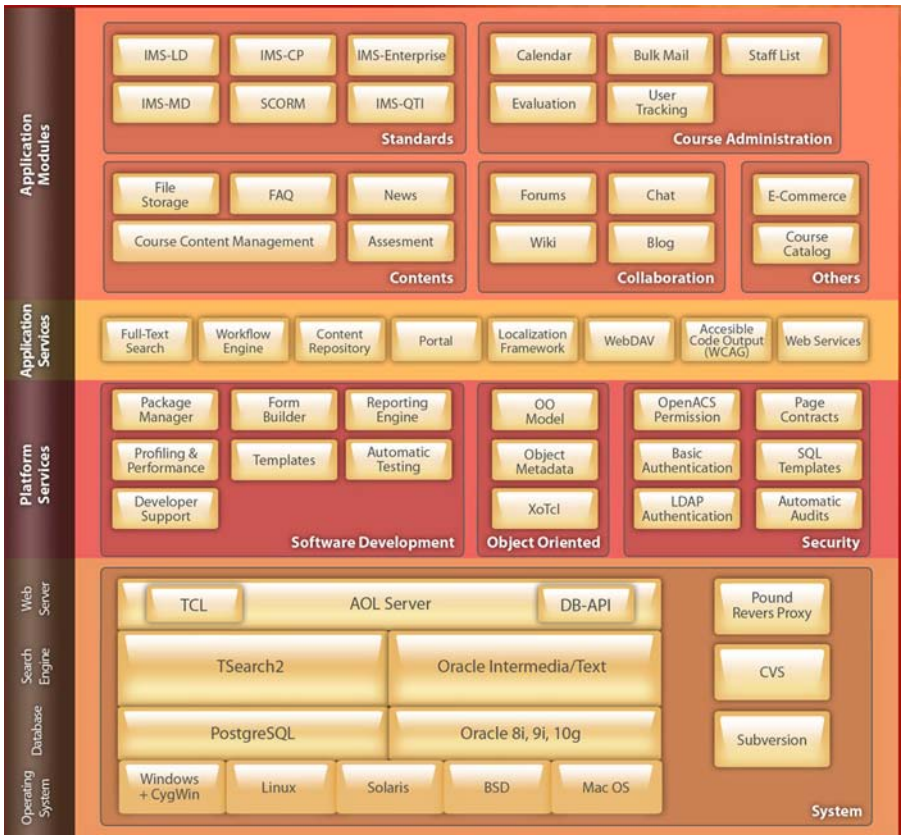


Figure 4.1: Complete architecture of the OpenACS platform. (Extracted from http://dotlrn.org/product/overview/)

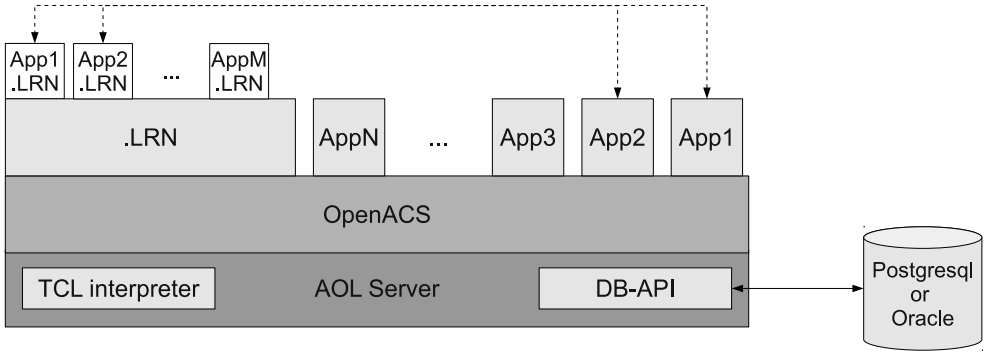


Figure 4.2: Simplified model of OpenACS and .LRN

The fact that .LRN is a community-oriented toolkit has influenced and shaped it into what it could be called a "communication oriented LMS". Most of the current LMSs focused at the beginning of their existence on providing content management for teaching staff and learners. .LRN, on the contrary, was conceived as a platform to facilitate communication among all the different actors in a learning experience. Just as an example, ever since the first release, each .LRN user has a Web folder shown in the login page to include both private as well as publicly accessible files. Each community of users has also its own area to exchange documents. Also derived from the community-oriented nature of the tool, there is a powerful user management model with rich functionality to handle groups of users and permissions.

Another feature of .LRN is its comprehensive notification mechanism. The underlying data model is object oriented, and each object may have a notification attached to it. Each time an object is modified, the notification information is processed and the proper email messages are sent. The user may choose to receive a notification whenever the objects in the platform are changed. This is especially effective for forum messages, shared files in a community, appointments, etc. The platform also allows each user to choose among instant, hourly or daily batched notification, providing a very effective mechanism to interact with the rest of the users.

But aside from these features, .LRN offers support for the most common specifications in *e-learning*. Course material can be uploaded in SCORM. The administrator uploads a zip file with a SCORM package and the system installs its content in the common file storage area of a class or a community. A portlet in the student area shows the links to enter a special screen to visualise its content as well as its index. This SCORM support offers the teaching staff to see the percentage of course material covered by each student.

Tests and quizzes are also fully supported in .LRN through the IMS Question and Test Interoperability (QTI) format [81], where the supported version is 1.2. Exam questions may be uploaded in this format as well as managed through the editing capabilities of the platform. Exams present inside a SCORM package are handled seamlessly by the tool by invoking the rendering engine and showing the content to the student. Teaching staff may manipulate exam content, results and statistics within the platform.

Open Source Community

Both platforms (OpenACS and .LRN) have GPL license. Their open source nature allows the code to be downloaded and modified. The documentation includes tutorials on how to develop new applications, some of which will be available from the official CVS. Apart from the CVS distribution of the code, a stable release of .LRN is regularly released. Current .LRN version is 2.5[2].

As in most open source projects, there is a community around .LRN/OACS involving more than 13,000 registered users. The community portal is itself based on this platform and coordinates the interaction between developers, users, technical personnel employed by higher education institutions and anybody interested on exchanging ideas, solutions and information about the tool. Also, documentation is available in wiki format, with regular updates on the content.

Users

The following institutions are a reduced sample of the type of environments in which this platform is currently being used (see [100] for a more detailed list including case studies):

[2]Checked on 2011-01-22

Harvard Univ. Executive Education Project. An *e-learning* platform was required to provide the best combination of flexibility, enterprise-class foundations, strong user base, and cost-effectiveness. After deployment, the project director acknowledged that .LRN provided "a huge head start toward what we wanted, plus support for things we didn't originally anticipate".

University of Heidelberg. The platform was chosen due to its licence that reduced the total cost of the virtual campus; and its flexibility, which helped on the development of case-specific features. Currently, the system has 30,000 members, where an average of 3,000 daily users is supported.

Vienna Univ. of Economics and Business Admin. It is one of the largest .LRN instances. It serves around 20,000 users, contains 26,000 learning resources, and averages 600 concurrent connections.

Universidad de Valencia. This university required a platform to support its conventional teaching for 40,000 users. After surveying platforms such as Moodle, ATutor, WebCT and ILIAS, their choice was .LRN due to its combination of scalability and extensibility.

Spanish National University for Distance Education (UNED). Both research and production are the reasons why this institution chose .LRN. The aDeNu research group have integrated adaptability and accessibility to the system. On the other hand, currently hosting of the virtual campus is providing service to 55,000 users with an average of 4 million transactions per day.

4.2.3 Implementation issues

GRAIL (Gradient-lab RTE for Adaptive IMS LD in .LRN) was the first implementation of the IMS LD specification fully built as part of a LMS. Considering Figure 4.2, the tool was conceived to be used in the context of a community. Consequently, GRAIL is implemented as an OpenACS module while the user interface is only offered to .LRN users. Some of the player requirements match with the existing LMS functionalities, where their reuse facilitates the implementation process. Figure 4.3 emplaces GRAIL in the .LRN architecture.

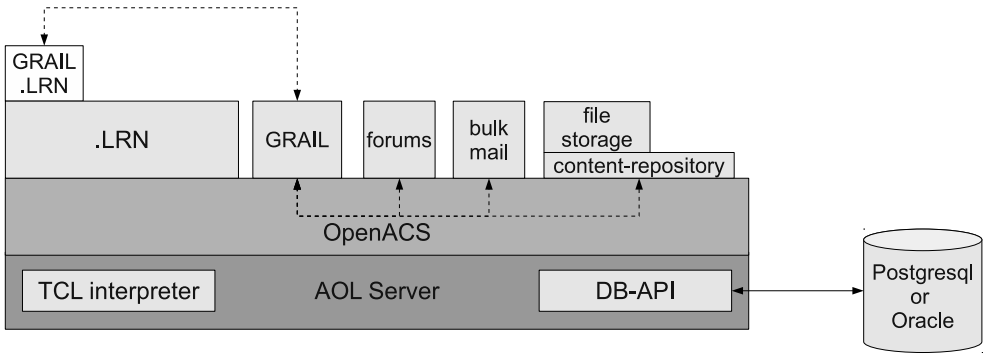


Figure 4.3: GRAIL on the simplified architecture model.

The community orientation of .LRN influences GRAIL so that UoL instances (called runs) are accessed from a community and the participants are selected among its members. A user in IMS LD is directly mapped to a user in .LRN. Therefore, all users within a

community (with their permissions) are automatically considered as potential users of a UoL. The attachment of users to a run is described in Section 4.2.5.

A significant part of the IMS LD specification is based in the use of roles; therefore, role management is one of the most relevant issues to be addressed when implementing a runtime environment. The .LRN platform contains a user/permission model which lent itself nicely to the implementation of role management. A role may have several role instances derived from it. These instances are mapped to the concept of sub-groups within .LRN. Group management also considers nesting, thus offering the perfect match to implement role nesting within the users of a UoL.

The object-oriented data model of OpenACS provides versioning capabilities to all objects. When the UoL is imported by the player, resources are stored as objects and therefore support new revisions. The design decision was to make use of the versioning system to simplify future improvements of the player, such as edition features. Such capability was used to provide runtime flexibility to courses, as described in Sections 4.3.3 and 4.3.4.

As described in Section 3.3.2, services are tools whose behaviour should be interpreted by the deployment platform. The supported services by GRAIL are *conference* (asynchronous and announcement types), *send-mail* and *monitor*. Support for *index* service was declined due to the lack of real use of this element in existing examples.

Functionality of communication oriented services, *conference* and *send-mail*, was reused from existing OpenACS modules: *forums* and *bulk-mail* respectively. In particular, the functionality to create several forums within a community offers the perfect underlying support to implement the *conference* service: when created in .LRN, forums are only visible to the users of the community in which they are instantiated. Since GRAIL is used in the context of a community, only people enrolled in it are able to join the discussions. These forums may be used with different management policies. Each role must then be able to join the forum with a different set of permissions. The user types included in the specification are: observer, participant, conference-manager and administrator. All of them are mapped into .LRN existing permissions.

Whenever a run of a UoL containing a forum is created, a new forum instance is also created. At this point, no user has access to the service, therefore it is invisible. When the users reach an activity where the forum must be used (that is, the environment contains the asynchronous *conference* service) their role membership is checked and the right permissions are assigned to the forum becoming visible in the user space (as any other forum in .LRN) even once the course has finished. Although not part of the specification, forums in .LRN support some extra functionality, for example, email notification for new posts, moderation, etc.

The level B of the specification defines the behaviour of the *imsldcontent*. An *imsldcontent* resource is a XHTML document containing elements to visualise and obtain a property or a group of property values. These elements are included in the document through the use of a different XML name-space. The run-time environment needs to detect the presence of such elements and perform two types of operations. If the element is of the *view-property* type, its content must be replaced by the current value of the specified property. If, on the other hand, the element is of type *set-property*, its content needs to be replaced by a form in which the user may introduce a value which will later be assigned to the given property. Also, these documents may contain elements with condition-controlled visibility classes. Although these classes are not part of the IMS LD specification, a mechanism needs to be implemented to deal with them.

The approach followed in .LRN consists in modifying the stored resource and creating a temporary file which is then delivered to the user browser. This processing is done entirely by the server. GRAIL manages this task by using the tDOM library, which offers a very efficient DOM implementation [179]. This tool provides an efficient environment to transform

imsldcontent elements into its correct XHTML to be delivered to the users.

An alternative for this server based approach using tDOM would be to delegate this task into the browser and use the Extensible Stylesheet Language Transformation (XSLT) or JavaScript DOM manipulation. However this solution would have browser compatibility problems difficult to solve (e.g. XSLT support, or the use of unsupported JavaScript libraries) and would increase the user requirements, so the decision was taken to perform this process entirely at the server side. Furthermore, in a client-side approach, the browser is responsible for hiding or showing content. This means that all the content is delivered to the browsers and experience users could view content when they were not supposed to.

The specification describes how conditions evaluation must lead to a newly calculated activity tree containing only those resources visible for a user and/or a role. Properties and conditions may be related with arbitrarily complex dependencies and, when a property value is changed, all conditions referring to it need to be evaluated. In GRAIL, the runtime environment stores the dependency relationship between properties and conditions. The initial value of all conditions is obtained when a new run of the UoL is instantiated. From that point on, when a property changes its value, only the related conditions are re-evaluated.

At parse time, conditions are stored in the database as defined in the UoL description, with their XML code. Condition evaluation amounts to parsing this XML replacing properties by their corresponding values. This scheme offered a simple implementation at a negligible computing cost and facilitates a hypothetical edition of conditions when an error is detected.

4.2.4 Integration with additional .LRN tools

The integration of IMS Learning Design and SCORM is still a pending issue discussed in Section 3.5.1. At a basic level, it is possible to take advantage of the combination of these two specifications. GRAIL supports the inclusion of a SCORM package as one of the resources of the UoL. Once it has been identified as a SCORM resource, the LORS OpenACS module is invoked and it takes care of the visualisation of its content packaging structure.

The inclusion of QTI resources, labelled as described in [150], is also managed by GRAIL: support for QTI through the Assessment package is offered in .LRN. Its treatment is similar to the one described for SCORM resources: if an activity includes a QTI-labelled set of questions, the rendering engine is invoked and its result is shown to the user. The platform then stores the received answers and returns the proper feedback to the user (as stated in the QTI specification). Figure 4.4 depicts how interaction among IMS LD, QTI and SCORM is achieved in GRAIL.

The possibility of sharing a common space for documents and folder is available in .LRN through its package called *file storage*. Combined with the permission model, it allows users to share documents among different groups, communities, courses or even have their own private space. An usage of such facility is to store all the resources present in a UoL. Thus, course resources can be accessed both through the GRAIL interface and the folder-based interface of the file storage. To prevent users from accessing still unavailable content (e.g. its corresponding activity has not been reached by the user), resources are initially unreadable and the right permissions are committed when the corresponding activity is reached.

4.2.5 The course life-cycle in GRAIL

GRAIL was conceived and developed as a IMS LD player. That is, the course life-cycle phases considered by this tool are deployment and enactment. Course authoring is delegated to the existing tools able to produce packaged UoLs such as Reload or Collage (see

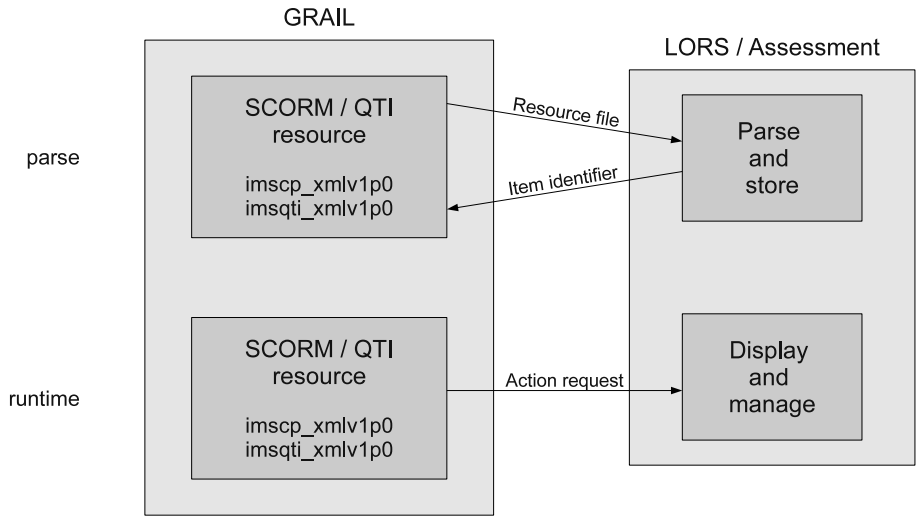


Figure 4.4: GRAIL management of QTI and SCORM resources.

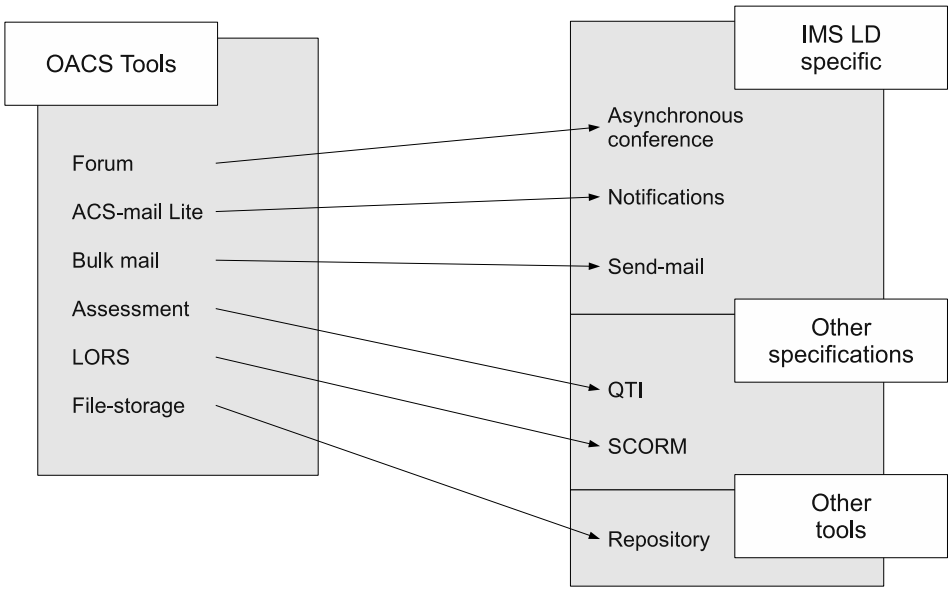


Figure 4.5: GRAIL interactions with existing .LRN services.

Section 3.3.3 for more authoring tools).

As well as other .LRN tools, GRAIL provides two user interfaces: the administration tool and the end user's access. The end user's interface is devoted to support the enactment phase, while the administration interface is used in the deployment and enactment phases. In other words, it is the course administrator who performs the UoL deployment. In .LRN, the teacher is who usually holds the administration privileges. For the purpose of this subsection, the terms teacher and administrator are indistinctly used.

GRAIL reuses .LRN functionality and stores the resources from the imported UoL into the LMS file storage, which in turn uses the content repository provided by OpenACS. The use of this existing functionality provides several advantages:

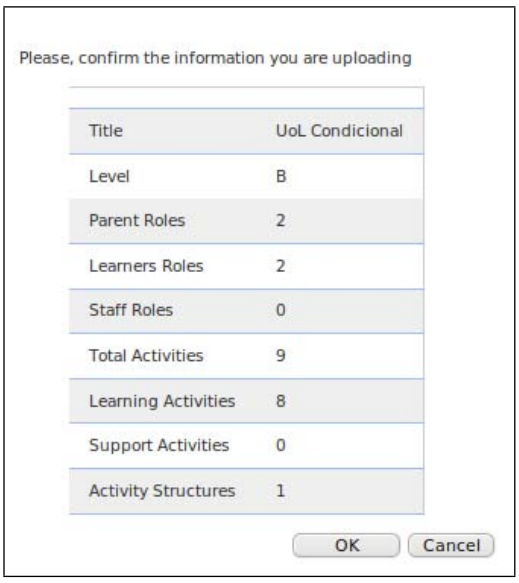
- New revisions of the material can be easily uploaded while older versions are not deleted.
- Resources are provided with a granular permissions system.
- The material can be accessed through the GRAIL user interface or using the folder view of the file storage.

Figure 4.6: IMS LD course administration page.

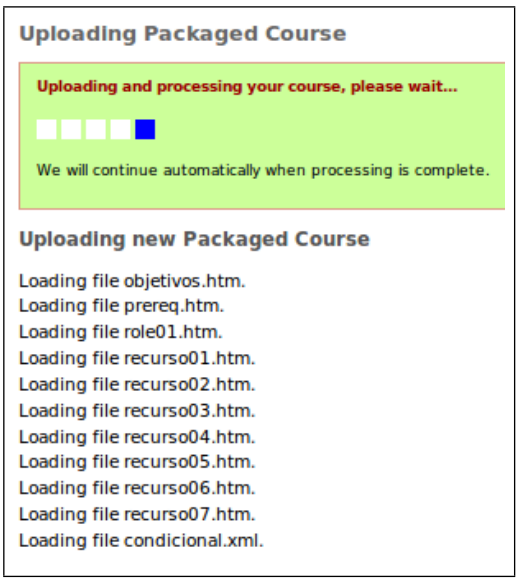
The first step to operate IMS LD courses in GRAIL is to import the Unit of Learning zip file. The tool then uses the tDOM library to validate and parse the content of the manifest file. If the manifest is correct, the administrator is prompted to load the elements of the manifest in the database thus creating a so called *Packaged Course*. A packaged course is the representation of a UoL loaded in GRAIL that needs to be instantiated in order to be enacted. When a Packaged Course is instantiated, then an enactable *Course* is created and showed as in Figure 4.6.

The relationship between a Packaged Course and a Course could be expressed with the analogy of object oriented programming, where the Packaged Course is a *class* and the Course is the instantiated *object*. The steps required to create a runnable course in GRAIL are depicted in Figure 4.7.

(a) The first step is to upload the package file.



(b) The administrator is prompted with information.



(c) The manifest is parsed and resources loaded.

Figure 4.7: Steps required to deploy and instantiate a Unit of Learning.

Once instantiated, a course can reach one of the four possible status: *waiting*, *active*, *finished* and *deleted*. The course starts in the *waiting* status, which means that the course instance exists but the participants have not been registered for the course. When the course has been populated, it becomes *active* and the participants start interacting with course material. The *finished* status is reached when the completion condition is satisfied. When a course is in the *finished* status, the course material is still available but the engine does not track the interactions. Apart from these three statuses that are defined in the specification, GRAIL introduces the possibility to set a course into the *deleted* status, with which the course is removed from users' interface.

The instantiation of a *packaged course* results in a *course* that needs users in order to become *active*. The GRAIL player is conceived to be used in the context of a community of users and these are the users who are candidates to participate in the instantiated *course*. That is, the members of the community in which the UoL is imported are the potential course participants.

Depending on the role structure and the number of students in a course, the task of creating role instances and assigning users to roles can be confusing. Providing an intuitive interface for this task was identified as an important requirement for GRAIL. Figure 4.8 shows a screen capture of the user interface for this task, where role management is presented as a matrix where columns are the roles and the students are the rows. The course administrator just has to check the cells to establish the relation between a user and a role. This interface hides the concept of role instances to the end user and provides a more comprehensive interaction. There also exists the possibility of using an interface that better fits the specification terms. The advanced interface (Figure 4.9) shows restrictions for role creation such as maximum or minimum number of students per instance. This information is obtained from the UoL description.

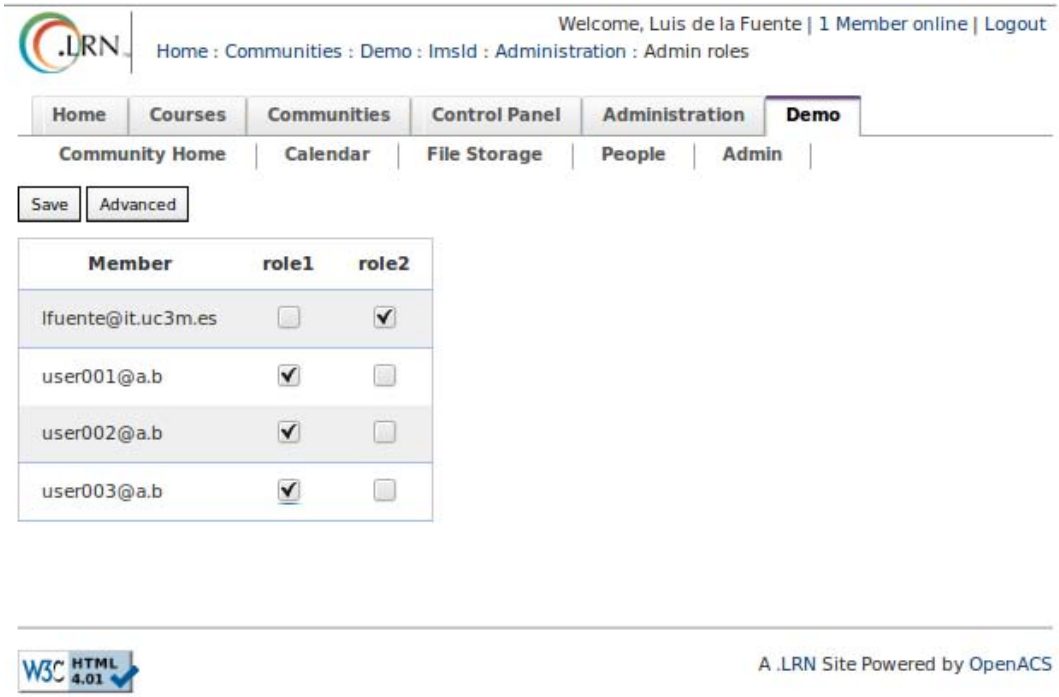


Figure 4.8: Role administration page.

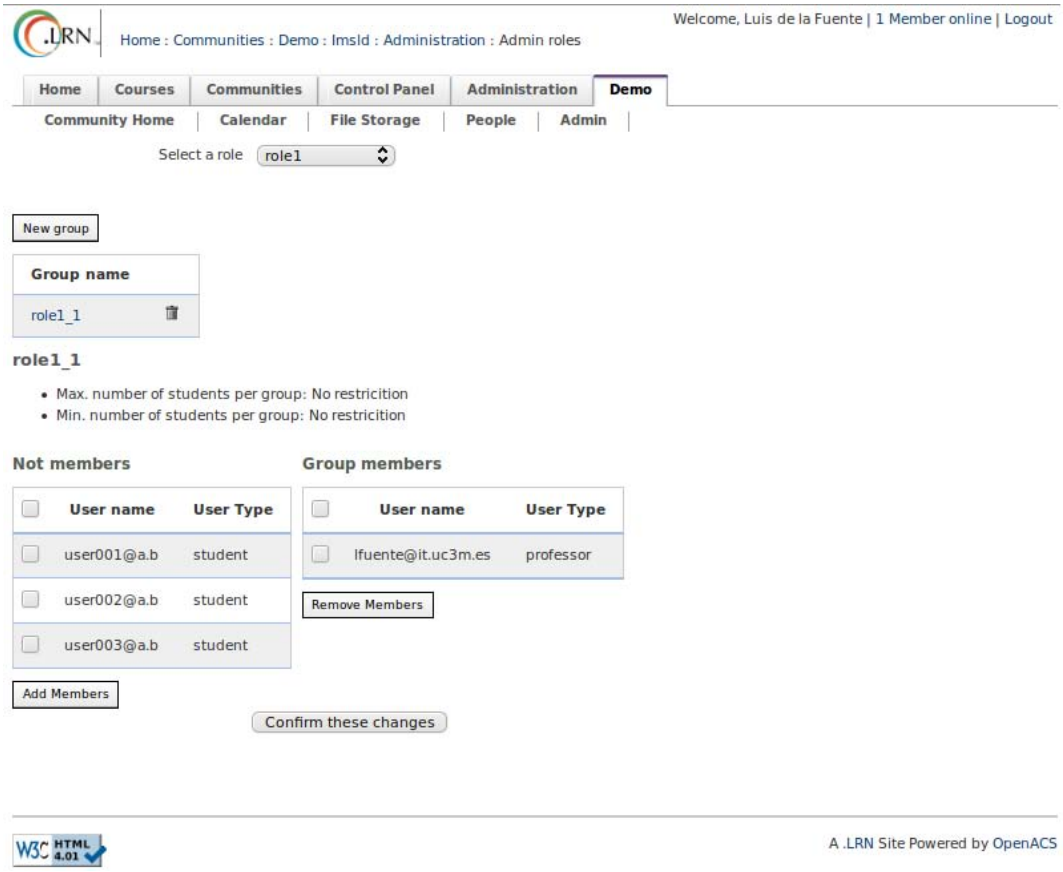


Figure 4.9: Advanced roles administration interface

Once the course reaches the *active* status, it can be accessed from the user interface. The course visualisation divides the screen in three different areas: the activity-tree, the related links and the material. The activity-tree is a set of structured links to the course activities. When a new activity becomes available, then the activity-tree is refreshed including the new link. When course participants click on a certain activity, the activity description is shown in the *material* area, while the *related links* area presents the set of links that, according to the environment associated to the activity, contextualized the realization of the activity.

A screenshot of the typical course visualisation is shown in Figure 4.10. There is a clear separation among the layout and the content of GRAIL's interface, where the layout is determined by CSS stylesheets and the content is provided by a carefully tagged HTML code. The benefit is the possibility of customising the layout on its very detail by just using a different CSS stylesheet, which is already supported in GRAIL.

The administrator's interface provides features beyond the course enactment. Among them, the most relevant one is the so called *cockpit*, a set of tools where the teacher is allowed to track students and edit certain aspects of the course. For example, it is possible to include new activities or modify their completion conditions. These features provide some flexibility to IMS LD courses that is discussed in Section 4.3.

In case that the practitioners want to use in a different platform a course that has been modified with GRAIL, the export functionality allows to obtain a *zip* file of the UoL. If the course has not been modified, then the exported course will be identical to the initially

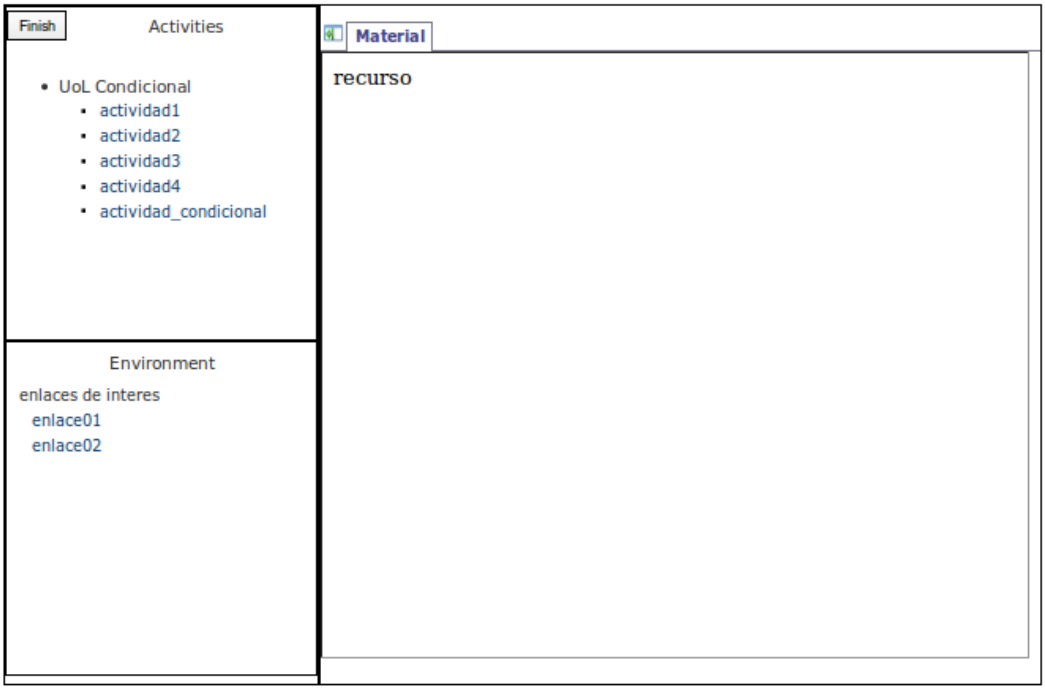


Figure 4.10: Course view in GRAIL

imported file. On the contrary, the exported package of a modified course includes its changes an also complies with the IMS LD specification. With the editing features and export capability, the linear structure of the course life-cycle presented in Figure 3.2 turns into a real cycle (see Figure 4.11). The feedback obtained during enactment allows course authors to improve the course material for future course replicas.

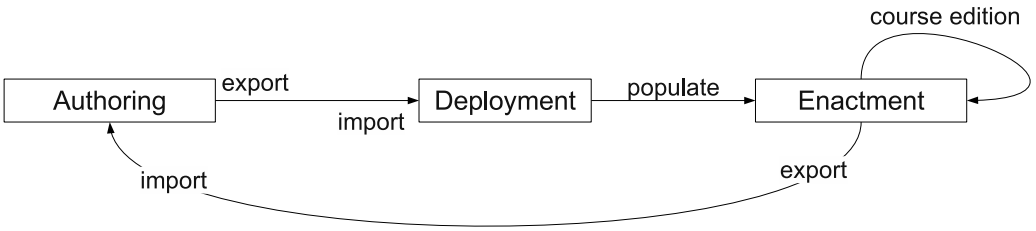


Figure 4.11: Course life-cycle with the edit and export features of GRAIL

4.3 Enactment flexibility

IMS Learning Design was built on the basis of reusability and interoperability and allows the execution of complex pedagogical models. With these two assumptions, the specification was

expected to be widely adopted by the *e-learning* industry but, years after its publication, these expectations have not been accomplished. One of the reasons of this low impact on real scenarios is the lack of the flexibility promoted by the model. Despite it is not imposed by the specification, current supporting tools favour the division of the course life-cycle in three isolated phases. The result of this division is a lack of flexibility that hinders the adoption of the specification.

The analysis of flexibility in the context of IMS LD and the development of solutions are presented as part of this dissertation [180]. The proposed solution allows instructors to react to unexpected situations in a way that the pedagogical model of the course is not affected by introduced changes. Flexibility was included as a GRAIL feature and has proved its usefulness in experiences such as [181, 182, 183].

4.3.1 Definition of flexibility

There is no agreement of the meaning of flexibility in the context of learning. In several works, the term is related to the possibility the platform offers to be used anywhere, anytime [184]. Thus, a course is said to be flexible if it allows course participants to interact with peers in with no place or time restrictions. The benefit of flexibility defined as such is that the potential range of lectures is broader, and that conversations can continue over the time. This definition of the term is more related to ubiquitous computing (then, ubiquitous learning) than to an approach to course flow modelling.

Another use of the term is given in [185], where the authors claim that the students follow different strategies, have different motivation and prefer different content types. Through the study of the student's behaviour, they concluded that each of them require to be supported by different materials. That is, course flexibility is used as a synonym of adaptability, where a flexible course delivers different material or activities depending on user profile. Other authors also use the term in the "adaptive" sense ([186, 187]).

Pierre Dillenbourg studied the term in the context of scripted learning flows. According to him, flexibility impacts the course life-cycle at its different phases [74]. On the one hand, course authoring is flexible if the scripting language provides means to express the instructors' hope. IMS Learning Design claims to accomplish enough authoring flexibility so that it is able to capture many different pedagogical models with the same vocabulary. The specification summarises this feature with the term "pedagogical neutrality". On the other hand, there is a difference between the interactions expected to appear in a course (the planned script) and what happens once the script is deployed and enacted (the actual interaction pattern). Flexibility is then related with the options that course participants have to tackle with this distance. The IMS LD specification does not provide any mean to reach enactment flexibility, nor restricts players to do so. In other words, the players are responsible to provide flexibility to IMS LD courses.

In the rest of this section, flexibility will be used as in the last definition: the options that the enactment platform provides to modify course characteristics and then tackle with unexpected events.

4.3.2 The need of runtime flexibility

There are several reasons for including flexibility as part of scripted learning courses. First, the risk of over-scripting: during authoring, it is almost impossible to find the threshold from which a higher degree of coercion results in a lost of efficiency of course material. Moreover, different scenarios may have different thresholds for this parameter. For example, students in higher education tend to be more self-taught when they are on higher courses. In such sense, the same pedagogical pattern would require a different degree of coercion for

being applied for freshmen or for experienced students. Flexibility of course material allows adjusting the constraints so that they adapt to the actual scenario.

In general, a course that is being enacted needs some degree of adaptation in order to be adjusted to actual scenario parameters. In collaborative learning, for example, the success of an activity usually depends on groups having the right number of components and being composed of the right user profiles. If the number of students does not match with the required situation, which actually happened in the experience described in Section 7.4, the course should allow the activity to be substituted by a different one with the same - or similar, at least - pedagogical objective.

Being the course adapted to the actual scenario, there is still room for unexpected events. On the one hand, statements may lack of clarity or even contain mistakes. In such case, the tutor would need to make the text more understandable, so he/she will need to rewrite the content. On the other hand, there are a countless number of unpredictable events. For instance, an activity may require being postponed due to illness of participants, technological problems or bad performance in a given session. Therefore, flexibility allows teachers to tackle with such unpredictable situations that may affect the efficiency of the course.

According to [74], teachers are responsible of the changes performed during the enactment phase. Therefore, they should be conscious of what can be modified and what cannot. Some constraints of the course are essential to accomplish the original pedagogical intentions and they should not be overruled. Constraints are classified in two different types: intrinsic, which are bound to the core mechanism of the script; and extrinsic, which are imposed by other factors like involved technology. In IMS LD driven courses, these restrictions translates in the following possible changes:

- Changes in material: Modifications like rewriting a paragraph, modifying an image or including further explanations about a topic are the most frequent required changes and it is unlikely that they broke intrinsic constraints.

- Changes in course flow: Structural modifications such as including, moving or deleting activities require to be careful, since these changes may affect termination conditions of the overall flow.

- Conditions edition: This is the most problematic type, because changes may introduce bugs in the course (infinite loops, unreachable states) which are not desirable in a live course.

Solving the above flexibility drawbacks in IMS Learning Design is still an open issue to be addressed by compliant tools. The next two sections are devoted to explain GRAIL's improvements on this area.

4.3.3 Flexible learning flow

A compliant player that covers all IMS LD features will allow enacting the ready-to-run packages loaded by the users. In these courses, teachers are enabled to track students' progress, but only to the extent provided by the monitor service if it is included in the course. The tracking facilities of this monitor service are based on the properties: teachers can track students as long as they can view what their property values are. The creation of monitor services requires a deep understanding of the specification and, as well as the rest of the course, is limited to what course authors decided to support.

In GRAIL, the administrative functionality has been incremented with an interface referred to as the cockpit. This cockpit implements the functionalities of the monitor service, and extends this service with tracking and edition capabilities. Unlike the monitor, the

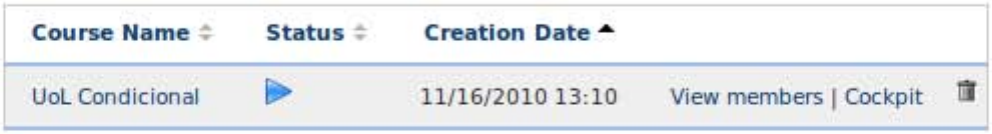


Figure 4.12: Link to the cockpit enabled for instantiated courses.

cockpit is not included during authoring and every course running in the platform has their corresponding cockpit. As shown in Figure 4.12, it is accessed through a link that is enabled when the course gets the *active* status. The features of the cockpit are described as follows:

Management of property values

The monitor service, as defined in IMS LD, requires an *imsldcontent*-typed resource that explicitly indicates what properties are going to be tracked. It is also required if the teacher should be able to read or write this value, with the *view-property* or *set-property* elements, respectively. There are situations in which the course author did not consider a property value to be relevant, but the enactment reveals that it actually is. To tackle this situation, the cockpit allows all the properties in the course to be administered.

For representation purposes, properties are classified depending on their scope, as depicted in Figure 4.13. *Global* and *local* properties are shared for all course participants so the property value is the same for all of them. On the contrary, *global-personal* and *local-personal* properties are instantiated for each course participant, so their property values are not necessarily the same. The cockpit contemplates this characteristic and, when personal properties are selected, allows indicating to whom the shown properties belong (see Figure 4.13). Similarly, *local-role* properties are replicated once per each existing role, and the cockpit prompts the administrator to decide the role to be tracked.

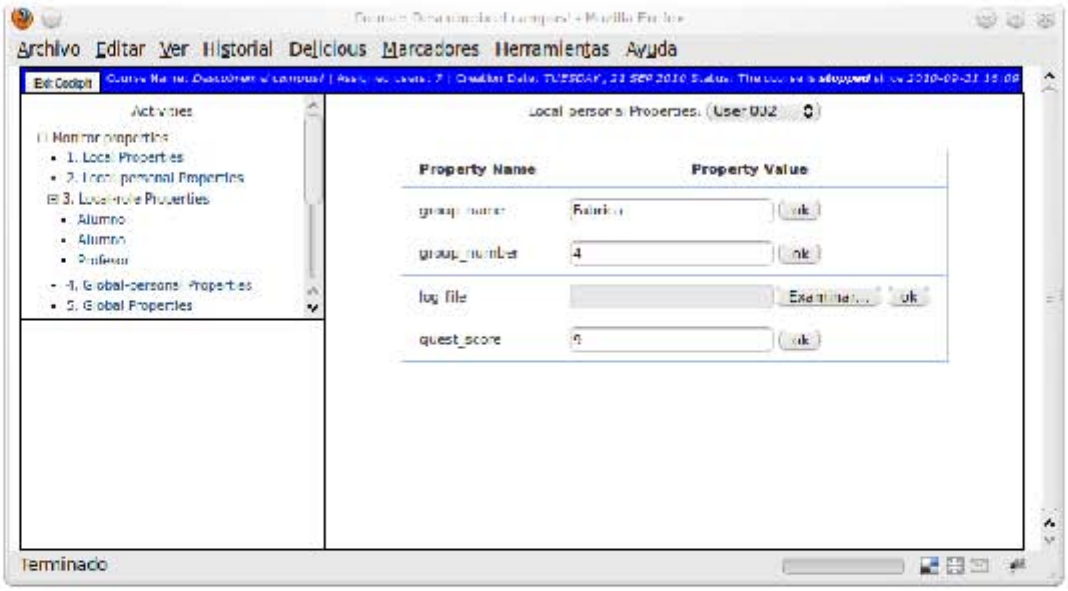


Figure 4.13: Properties representation in the cockpit.

Each property generates a form whose initial record is the property value. Thus, the property can be read and modified by simply changing the value and pressing *OK*. As with properties in the course content, the modification of a property results in the re-evaluation of the corresponding conditions. This feature enables teachers to act on behalf of students if the situation suggests so (as in the case of the experience in Section 7.4), remove unfair[3] penalties imposed to students or manually adapt the course flow (as in the case of the experience described in Section 7.3).

User tracking

The IMS LD specification allows creating fine-grained learning flows and to impose conditions to the different paths. It does not provide, however, a method that allows the teacher to track which student is following each path. Again, all the tracking facilities rely on the monitor service, whose expressiveness is insufficient for the purpose. The cockpit in GRAIL provides two types of reports that support the teacher while tracking student's progress.

The first of the reports is activity-centred. That is, presents a table (presented in Figure 4.14a) with all the students who have accessed an activity. More specifically, the information shown for each student is the first access to the activity and, if the activity has any completion condition, the timestamp of the completion event. The second type of report, shown in Figure 4.14b, is student-centred: presents a table with all the activities ever accessed by a student, with the corresponding first access and finalisation dates.

The use of activity reports is especially relevant when is the teacher who leads the pace at which the students work, so he/she can wait for all group members to have finished certain activity to activate the next step. In collaborative learning flows, this reports help to identify which of the teammates is more active on the course, or if a certain student is getting the benefit without developing his/her corresponding task.

Learning flow modifications

Under certain circumstances, the cockpit allows course administrators to add activities to, or delete from, the learning flow. These particular circumstances are:

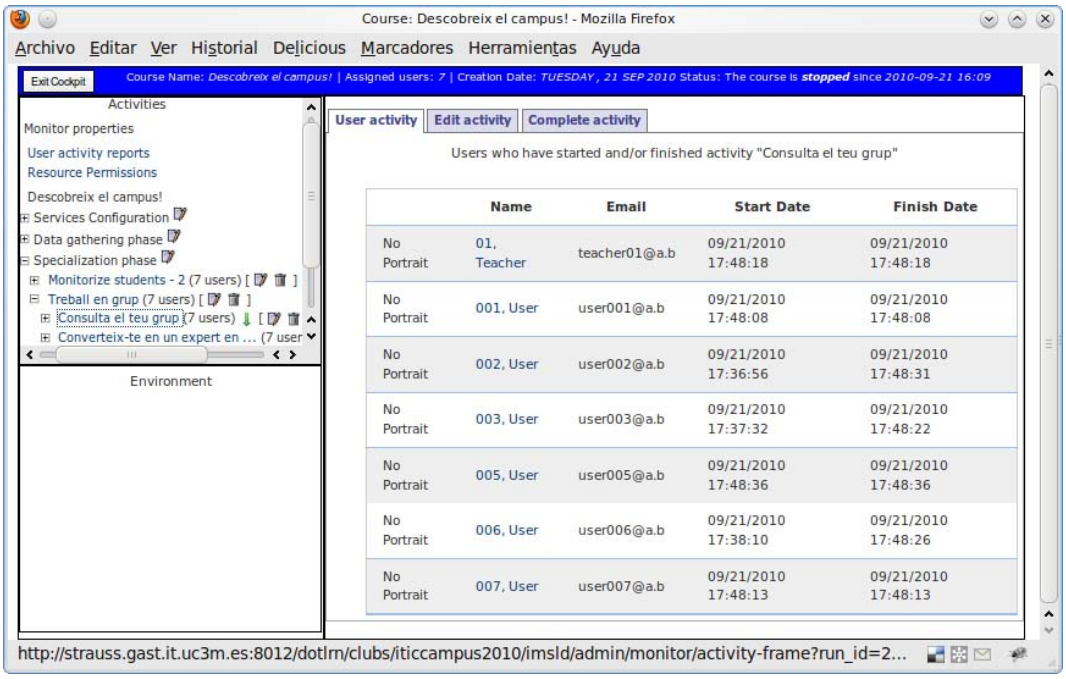
- The activity is part of a structure. This condition was imposed because, given the GRAIL implementation, the addition of activities out of a structure was difficult to implement.

- The act in which the activity is going to be added has not been yet enacted. This condition prevents for unstable course states to appear.

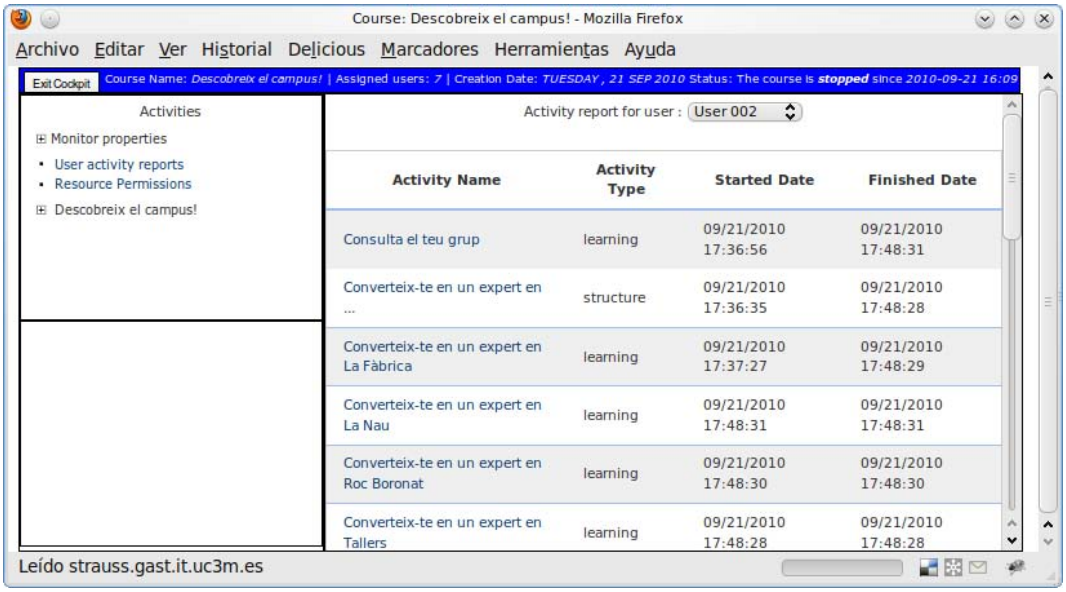
It is also possible to modify existing environments by adding, editing or deleting their learning objects. Since these resources or services can belong to an external entity, there is a chance for them to change, which would significantly affect the course intrinsic constraints; by having the ability to manage these items, the learning staff is able to tackle issues like the one described.

The addition of new learning objects to the environment and activities in the learning flow also allows the course to be "alive": for example, the tutor can incorporate yesterday's news as readings, which possibly engages students more than older news would do. This functionality also allows teachers to react if the course context suggests a new activity to be included.

[3]For example, a student who was sick and whose absence was punished



(a) Activity-centred report.



(b) Student-centred report.

Figure 4.14: Cockpit reports for students' tracking.

Conditions establishment

As termination condition for an activity, IMS LD allows to specify a period of time after which the activity is supposed to be marked as finished. It is not realistic to expect this condition to be always applicable. There could be, for example, a holiday period that forces the activity to take longer than expected. In these cases, the course constraints require the run time engine to provide certain degree of flexibility.

GRAIL allows modifying the termination condition of any activity so the course can be adapted to the actual scenario. The interface for completion conditions edition is shown in Figure 4.15.

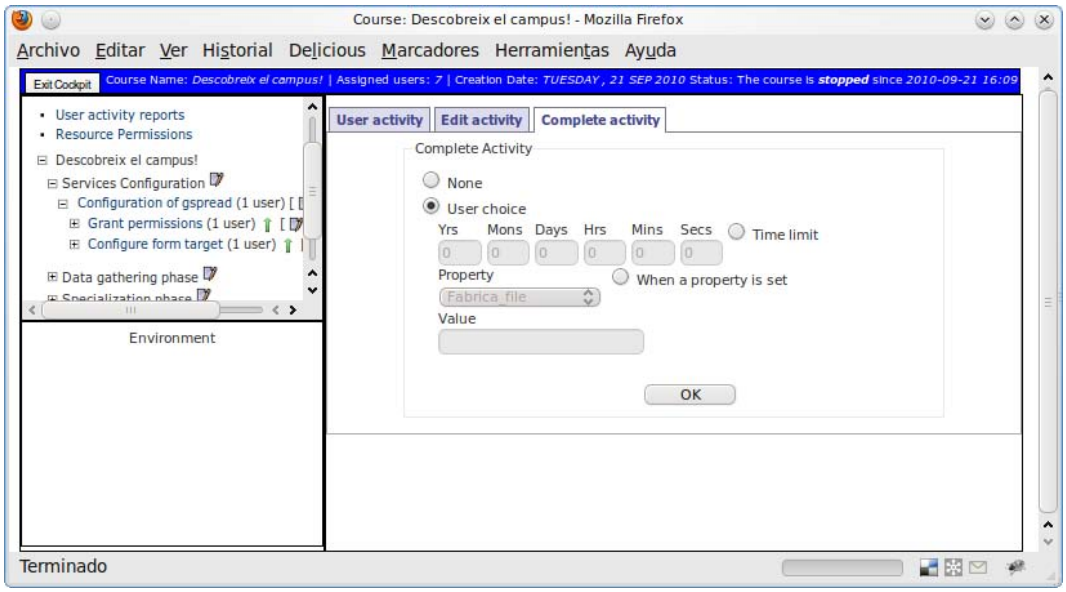


Figure 4.15: Interface for the edition of completion conditions.

4.3.4 Flexible content

Content in IMS LD usually takes the form of Web content, i.e., any content that can be rendered by a Web browser. The most common used format is HTML, but there is still room for multimedia content in the shape of flash applications, Java applets, images, videos, etc. According to the *de facto* course life-cycle, content is created during the authoring phase, and delivered at the enactment. However, as discussed in Section 4.3.2, it is very likely for the content to include mistakes or inappropriate material. Such problems commonly arise during enactment, when the material is no longer modifiable.

GRAIL provides two different means to update course material once the package has been imported and the course is being enacted. It is up to the course administrator to select which of the allowed methods are preferred. We will refer these methods as the *file-storage* and the *xoWiki*.

The *file-storage* method consists in a simple file substitution. As viewed in Section 4.2.3, GRAIL handles the UoL resources with a repository that can be browsed through a folder-view. Thanks to the versioning system offered by the OpenACS toolkit, content files can be replaced at the time that they keep their unique identifier, so the IMS LD player does not

perceive any difference. The versioning system also allows recovering older versions of a file. This edition method was used in the experience presented in Chapter 7.3.

The simplicity of the *file-storage* method may be a double-edged sword. On the one hand, substituting the file with the new version is the only required action. On the other hand, the creation of the new version of the content require the use of an external authoring tool and poses the teacher to the need of knowing the filename that corresponds to a certain activity. In practice, file substitution requires to know the structure of the content, which is known by the course author but not necessarily by the teacher. As a consequence, the method is appropriate when the author and the teacher are the same person, but it is not so useful in other cases.

The other method to edit content is based on *xoWiki*, the wiki platform embedded in the OpenACS toolkit. The functionality can be summarised as follows: during the package import, the content is translated to wiki format, so this tool is in charge of storing and rendering the material. When the teacher needs to modify some content, he or she just access to the corresponding activity and clicks the "edit" button. The modifications are done in wiki format, whose simplicity does not require users to be trained.

The wiki allows performing modifications in the content without requiring previous knowledge of the UoL nor advanced computer skills. However, this is at the cost of the introduction of a new requirement: a permissions system. To understand this requirement, let's have an example: a UoL with several student roles but no teacher role is loaded in the platform. Then, when the course has been instantiated and populated, a flaw in the content is detected. Which role is allowed to edit the content? For the sake of the understandability, we have stated that the teacher is enabled to make changes but, is the teacher the only allowed role? What if there is no teacher role?

There is no unique solution for the above question. GRAIL makes use of the underlying permissions system so that write access is granted for roles and subroles of type *teacher*, while roles and subroles of type *student* have read permissions. The possibility of modifying these settings is provided within the cockpit, so that the course administrator controls who can and who cannot modify the content. Figure 4.16 shows the permissions manager in the cockpit.

Although it was not the aim of such development, the introduction of wiki behaviour on the UoL content opens new learning scenarios that were not possible without such functionality. For example, a teacher could intentionally include errata in course material, so the students are expected to find and fix them. There, the teacher has the opportunity to revise the wiki history, so he/she can see who did the change.

The content edition features, joined to the course flow edition ones, allow the runtime environment to behave like a collaborative authoring tool. In such scenario, the case of use could be described with the following steps:

1. With the help of an external authoring tool, a UoL with no content is created. Such UoL should contain the skeleton of the final course and should link to empty files used as resources.

2. The skeleton UoL is uploaded in GRAIL, and xoWiki is selected as content manager.

3. The course is instantiated and populated by the forthcoming authors, who access the activities and create the material.

4. When finished, the course can be either played in GRAIL or exported in a different UoL.

Such collaborative-authoring-tool scenario, as well as other wiki based activities, has not been tested with real experiences and it is therefore difficult to identify its drawbacks.

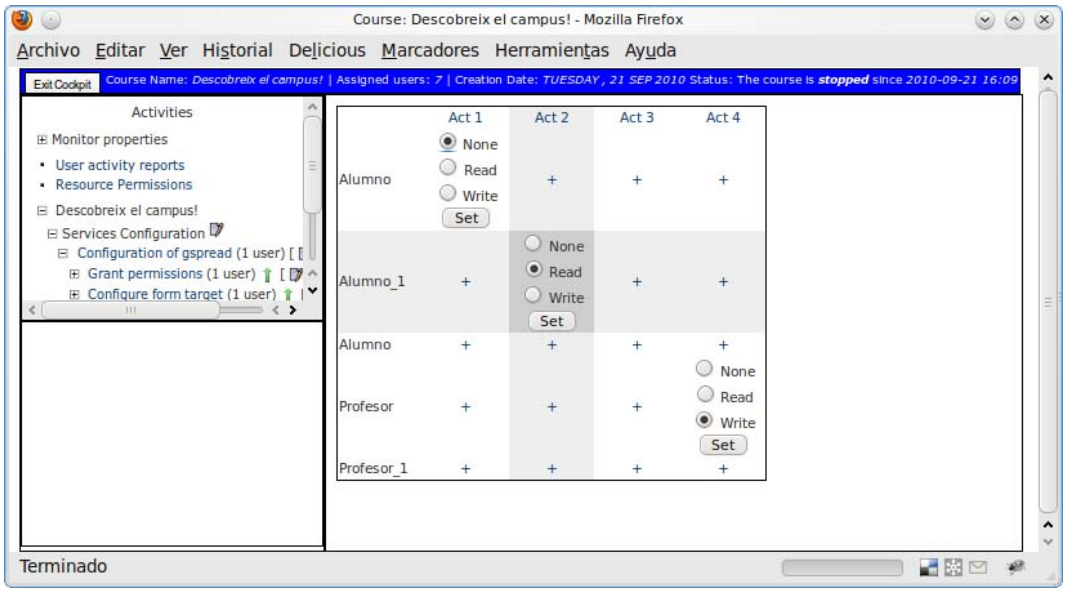


Figure 4.16: Permissions manager for UoLs that use xoWiki.

4.4 Conclusions

This chapter presented the issues that arose during the implementation of the GRAIL runtime environment of IMS LD, implemented as part of the .LRN platform. The development of a complete player from scratch is a difficult project that, having the availability of already existing players, required to be justified with forthcoming benefits over the alternatives. In the case of GRAIL, the high modularity of the Web toolkit that supports the overall platform promotes the quick implementation of extra functionalities through the reuse of existing modules. Such capability allowed integrating the IMS LD player with the typical functionalities of a community (file-storage, notifications, forums, etc.) and with the already existing support for *e-learning* specifications (SCORM, QTI). As a result, GRAIL is a IMS LD player that complies with the specification and provides extra functionality that improves the learning experience.

It is noteworthy the support of runtime flexibility that enable teachers to tackle with unexpected situations that may arise during the course enactment. The lack such flexibility is precisely one of the major limitations of IMS LD and prevent practitioners to adopt the specification for its regular use. GRAIL allows introducing modifications both in the course content and the learning flow, so that the course enactment has a fair enough flexibility degree. Furthermore, the use of the .LRN wiki capabilities for the content edition enables new scenarios such as the use of GRAIL as a collaborative authoring tool, or the use of the edition capability as a pedagogical resource in learning courses.

Chapter 5

Service integration in IMS LD courses

Talk is cheap, show me the code.

Linus Torvalds

5.1 Introduction

Our relationship with information and content has been radically shifted from the situation where the user is a content consumer, to one where the users' interactions produce and use this content. The scenario can be described with a Stephen Downes quote: "Web 2.0 is not a technological revolution, it is a social revolution" [23]. Marc Prensky introduces the term *digital natives* [188], and asserts that attention and learning skills of users who have grown in the computer era are different shaped than in *digital immigrants*.

In this scenario where users' interaction prevails over other activities, it is desirable to take advantage of the influence of Web 2.0 tools and include their use in teaching/learning processes. As discussed in Section 2.2, this inclusion would enrich existing pedagogical methods, specially those based on collaboration and information sharing.

The simple execution of learning activities does not necessarily produces learning. In order to accomplish a complete learning experience, activities are required to be orchestrated with a pedagogical sense. Different orchestration methods are described in Section 3.6. Among them, the most promising one and the *de facto* standard for learning orchestration is IMS Learning Design, described in Section 3.3.

However, one of the common critiques to the IMS LD framework is the lack of integration of learning tools in the activity flow. A proper solution would offer a method to effectively integrate Web 2.0 tools in the context of IMS LD courses. The different proposed alternatives are exposed in 3.5. Among them, there is no solution that does not restrict the integration to a given type of tool, at the same time that allows exchanging information between the learning flow manager and Web 2.0 tools.

The main proposal in this dissertation is the definition of a framework that complements IMS LD so that it allows the integration of third-party tools (called, in this context, external services) without restraining the pedagogical neutrality, reusability, self-containment, adaptability and collaborative features of the orchestration method. The proposed framework is called *Generic Service Integration* (henceforth GSI) and allows the integration of web based tools that comply with some minimun requirements in the context of IMS LD.

GSI is a layer that mediates the communication among IMS LD and the external services. The architecture is based on its modularity, so that the integration of different services is realized by different service adapters easily plugged to the GSI layer. Service integration impacts the whole course life-cycle and introduces new requirements in courses enactment, specially devoted to authentication issues.

The rest of this chapter is devoted to introduce the GSI proposal. The first Section (5.2) gives an overview of the model and details how service integration is managed in the different phases of the course life-cycle. Due to its relevance on the integration, identity and authentication issues are discussed in Section 5.3. The model has been implemented as part of GRAIL, the IMS LD player in .LRN. The implementation details and the developed service adapters are presented in Section 5.4, as well as a step-by-step guide to develop new adapters. Finally, in Section 5.5 it is discussed how the model provides the functionality without restraining the pedagogical neutrality, reusability, self-containment, adaptability and IMS LD.

5.2 Generic Service Integration

The *Generic Service Integration* (GSI) framework allows the integration of Web tools in the context of IMS LD courses. The proposal is based on the IMS LD specification: GSI elements are included in the manifest and their inclusion impacts the complete course life-cycle. Thus, the orchestration of activities is done by the IMS LD server, while the activities can be performed in a different Web tool. In this chapter, integrable tools are referred as external services, third-party services or simply services.

The relevance of service integration in learning flows was discussed in [189] and an earlier version of GSI was published in [190, 191]. This section is devoted to explain GSI from different perspectives: first, an overview of the proposal captures the objective and work philosophy, then the details of authoring, deployment and enactment are presented.

5.2.1 General description

The goal of GSI is to allow the use of Web tools as part of a UoL. The Web tools to integrate can be provided by different vendors than the IMS LD player's provider. Any existing Web-based tool that offers a API is therefore potentially integrable in a UoL via GSI. The architecture and course life-cycle of GSI are defined so that UoLs maintain the principal characteristics of IMS LD, which are described as follows:

Pedagogical neutrality. IMS LD is potentially capable of expressing a wide range of pedagogical methods for learning and this is one of the strengths of the specification. The integration of services in IMS LD courses should not be restricted to a particular learning/teaching method.

Reusability. The "write once, run many" philosophy reduces the cost of orchestrated learning courses and increases their quality. In order to promote reusability, the course instantiation should be accompanied by automatic service instantiation,

Self-containment. All the information required to deploy and enact a UoL must be included in the course package, so the difference between two replicas of the course is reduced to the minimum. In the case of external services it is not possible to package the web server in the distributable zip file and services are included by means of their description.

Collaboration. As well as IMS LD provides roles to support collaborative learning flows, the integration of Web tools should encompass this feature and provide the means to transport the concept of role to the external service.

Adaptability. Properties and conditions support the development of adaptive strategies. The use of runtime information to adapt the course content requires the framework to provide bidirectional exchange of information with the external tool during the enactment phase.

Generic Service Integration is not implementation dependant. The proposal has been prototyped in GRAIL, it could be implemented on any IMS LD player where the only requisite is being connected to the Internet[1] in order to have access to the integrated services. In fact, GSI could be easily extrapolated to other contexts (such as Moodle-based course delivery) by transposing its ideas to the potential target framework.

Third-party tools are very heterogeneous: there exists domain-specific and general-purpose services that can be used for very different purposes. The information they host cannot be classified within a single taxonomy and there is no open standard followed by all of them. Therefore, it is practically impossible to develop a *plug and play* technology that connects all existing tools and that requires no adaptation for the particular cases. The approach taken in GSI is the use of service adapters that translate the particular API of third party tools to a format understandable by the IMS LD player. That is, each integrated tool requires the development of a case specific adapter. With the aim of avoiding an unaffordable methodology of service integration, one of the design principles in the development of GSI was that the implementation process of new adapters must be as simple as possible.

Service integration should not increase the difficulty of the course authoring process, which is recognized as one of the weaknesses of IMS LD (see Section 3.3.3. Therefore, the vocabulary provided to describe third-party tools in the manifest has been kept simple, even at the cost of providing less expressiveness. The inclusion of third party tools increases the difficulty of the course authoring phase but keeps it into reasonable margins, so that this increment is justified by the benefits provided by service integration.

The *Generic Service Integration* framework follows a "early-defining, late-binding" approach that impacts the complete course life-cycle: the requisites of the Web tool are defined in the course authoring, the deployment phase is complemented with the need of selecting the more appropriate tool, and the enactment allow human and programmatic interactions with the selected tool. Figure 5.1 depicts the GSI behaviour through the course life-cycle.

During the authoring phase, course authors may decide that one activity must be performed using a certain Web tool. Then, the course author creates an environment and includes a *genericService* as the element of this environment. The *genericService* is used to describe the characteristics that the required tool should offer. This description is given in a generic manner, without specifying a specific tool but specifying the requirements imposed to the tool: supported permissions, functionality, allowed methods, etc. When the UoL is being deployed, the *genericService* element is detected and the more adequate tool is selected among the available ones, which are the tools explicitly supported by means of an installed service adapter. This fact provides some flexibility degree to the deployment proccess, because different instances of the same course could be supported by different third party tools. For example, one tutor prefers the use of *Wikispaces* for the support of collaborative document creation, while another tutor may prefer *Google Docs*. Both of them can use their prefered tool and the course still holds its intrinsic characteristics. The external tool is then instantiated (e.g. an empty document is created) and the URL that links to the created instance is placed in the environment that contains the service. During

[1]Strictly speaking, IMS LD compliance do not require being connected to the Internet.

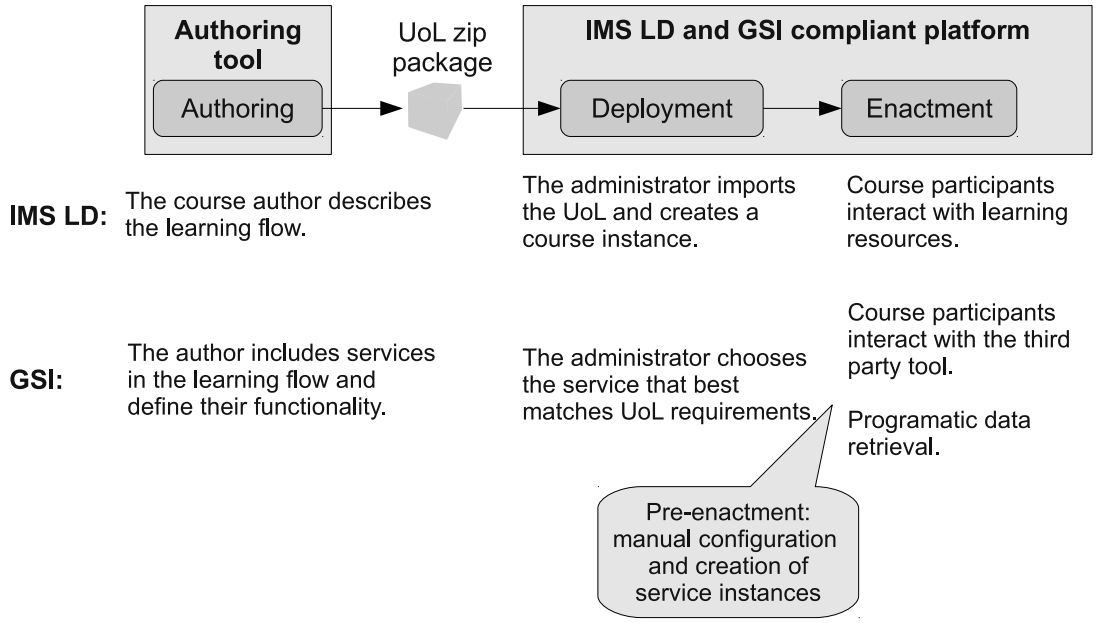


Figure 5.1: GSI behaviour through the course life-cycle.

the enactment, course participants interact with the service accessing this link, while the IMS LD player exchanges information with the external tool and stores it in properties so that they can be used in conditions.

GSI service adapters use the proprietary API (or whatever access method) offered by the external tool as a mean to access the service information. According to GSI, the external tool always takes a passive role in the information exchange and behaves according to the requests coming from the IMS LD server. The underlying idea is that the accessed service requires no modification to be integrated in the course, and the service adapters use the existing features to satisfy as many GSI features as possible.

5.2.2 GSI vocabulary

When the course author decides that a certain activity needs the support of an external tool, he/she usually thinks in a specific tool from a specific vendor. For instance, the use of Google Docs to collaboratively create a document. However, the author does not know the platform where the course will be enacted so he/she ignores the possible restrictions. For example, it may happen that an institution hosts tools that provide the same functionality offered by Google and internally promote the use of these self-provided tools.

In most cases, the unavailability of a certain tool can be overcome by substituting it with a similar one. In other words, the author is thinking on a certain tool but he/she really needs the functionality, not the specific vendor. For example, the functionality of Google Docs can be substituted by a different wiki provider without severely impacting the learning flow. GSI allows course authors to describe services by their functionality, at the same time that permits recommending a given vendor in case it is available.

Even without considering external services, course authoring in IMS LD is recognized to be a challenging task for practitioners (see Section 3.3.3) and it does not seem reasonable to increase the complexity of the process with the inclusion of new features. Considering this

recommendation and the above discussion, it is clear that the vocabulary provided by GSI should be versatile enough to define any type of tool and simple enough so that it does not increase the complexity of the authoring process.

The vocabulary offered by GSI has two main parts: the service definition and the establishment of a relationship between a property and the information from the external service. The latter is a simple XML element, while the former is a complex type with several sub-elements. The root element of such complex type is called ***genericService***. This subsection details the structure of such element and supports the explanation with fragments of the XML schema. Figure 5.2 depicts the datamodel of a *genericService*.

genericService

IMS LD defines the environment as "a collection of specific objects and services needed to perform the activity" [6]. In this definition, the term *service* refers both the tools defined by IMS LD (see Section 3.3.2) and any other tool that could be included to support the activity.

The code in Listing 5.1 is a fragment of the IMS LD schema. It shows that the *serviceType* (the element that is placed inside an *environment*) supports native services (i.e. *grp.service*) and the inclusion of any element from another namespace (i.e. *grp.any*). The *genericService*, the service type defined by GSI, is therefore placed inside an *environment*.

The *genericService* (schema code listed in Listing 5.2) is composed by six sub-elements, where the most relevant ones are *groups*, *tool* and *constraints* that respectively define **who** will take part in the service, **what** the required functionality is and **how** will it be used. The title and description are needed for runtime representation purposes. The *alternatives* element is a container of references to other services and/or learning objects to be used in case the enactment platform cannot select a service that complies with the requisites specified in the *tool* element.

groups GSI defines a *group* as a reference to exactly one IMS Learning Design role, as stated by the schema showed in Listing 5.3. That is, establishes who the user of the service is. Any other reference to service users made inside the *genericService* must refer to a *group* element, instead of a IMS LD role.

This redundancy between *groups* and roles aims at the reusability of service definitions. The idea is to keep at the minimum the inclusion of IMS LD elements in the GSI service description, so that the service can be easily reused in another UoL, or in another scenario not based on IMS LD.

```
1  <xs:complexType name="serviceType">
2      <xs:choice>
3          <xs:group ref="grp.service"/>
4          <xs:group ref="grp.any">
5              <xs:annotation>
6                  <xs:documentation xml:lang="en">This placeholder can be used
                       to import elements from another namespace. The namespace
                       for the imported element must be defined in the instance,
                       and the schema must be imported.</xs:documentation>
7              </xs:annotation>
8          </xs:group>
9      </xs:choice>
10     ...
11 </xs:complexType>
```

Listing 5.1: "Schema of IMS LD serviceType"

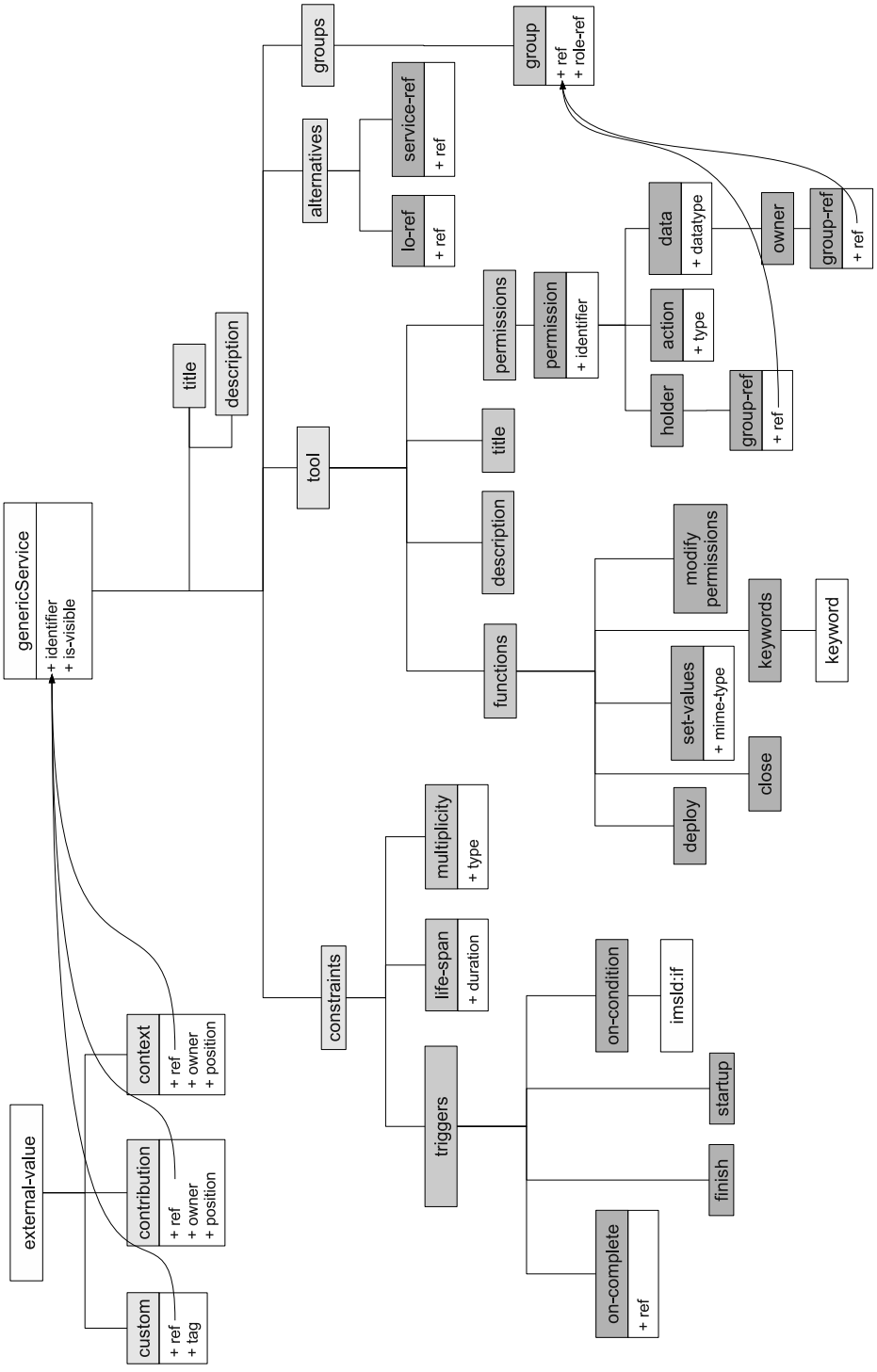


Figure 5.2: Complete GSI datamodel.

```
<xs:element name="genericService">
    <xs:complexType>
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs="0" ref="title"/>
            <xs:element maxOccurs="unbounded" minOccurs="0"
                ref="description"/>
            <xs:element ref="groups"/>
            <xs:element ref="tool"/>
            <xs:element ref="constraints"/>
            <xs:element maxOccurs="unbounded" minOccurs="0"
                ref="alternatives"/>
        </xs:sequence>
        <xs:attributeGroup ref="attr.identifier"/>
        <xs:attributeGroup ref="attr.is_visible"/>
    </xs:complexType>
</xs:element>
```

Listing 5.2: "Schema of genericService"

```
<xs:element name="group">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="title" minOccurs="0"/>
            <xs:element maxOccurs="unbounded" minOccurs="0"
                ref="description"/>
            <xs:element maxOccurs="unbounded" minOccurs="1" ref="ld-role"/>
        </xs:sequence>
        <xs:attributeGroup ref="attr.identifier.req"/>
    </xs:complexType>
</xs:element>
...
<xs:element name="ld-role">
    <xs:complexType>
        <xs:attributeGroup ref="imsld:attr.role-ref.req"/>
    </xs:complexType>
</xs:element>
```

Listing 5.3: "GSI groups definition"

tool The *tool* element describes the functionality that the external service should accomplish and includes the *functions* that should be supported by the service, the *permissions* that will be applied to the participants, and the human readable *description* (see Figure 5.2 and Listing 5.4). Course authors use the *description* to provide an explanation of the tool they are expecting. The *keywords* are a set of tags that completes the definition of the service and the deployment platform can use them to filter the available adapters during the tool selection.

The requirements specified in the *tool* element should reflect the optimal configuration of the service. That is, the activity was conceived to be performed with all the specified features, but a subset of them should be enough to successfully perform the activity. In the worst case, the selection of a service that does not acomplish with the required features could result in the loss of the intrinsic characteristics of the activity. A more detailed analysis of the implications of service selection are discussed in Section 5.2.3.

The *functions* element is used to specify what methods should be supported by the service. For the sake of simplicity, there are four allowed methods: *deploy*, *close*, *set-values*, *modify-permissions*. GSI provides guidelines of how these methods should behave, but the actual behaviour of the verbs is given by the service adapter. Section 5.2.6 provides more

```
<xs:element name="tool">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="title" minOccurs="0"/>
            <xs:element ref="description" minOccurs="0"/>
            <xs:element ref="keywords"/>
            <xs:element ref="functions" minOccurs="0"/>
            <xs:element ref="permissions" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
...
<xs:element name="functions">
    <xs:complexType>
        <xs:all>
            <xs:element ref="deploy" minOccurs="0" maxOccurs="1"/>
            <xs:element ref="close" minOccurs="0" maxOccurs="1"/>
            <xs:element ref="set-values" minOccurs="0"/>
            <xs:element ref="modify-permissions" minOccurs="0" maxOccurs="1"/>
        </xs:all>
    </xs:complexType>
</xs:element>
...
<xs:element name="permissions">
    <xs:complexType>
        <xs:sequence>
            <xs:element maxOccurs="unbounded" minOccurs="1" ref="permission"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
...
<xs:element name="permission">
    <xs:complexType>
        <xs:sequence>
            <xs:element ref="holder"/>
            <xs:element ref="action"/>
            <xs:element ref="data"/>
        </xs:sequence>
        <xs:attributeGroup ref="attr.identifier"/>
    </xs:complexType>
</xs:element>
```

Listing 5.4: "GSI tool definition"

details on how the service adapter can provide the concrete meaning to these methods.

- The *deploy* method is used to instantiate the service, returning an URL. A service that cannot be deployed is always accessed through the wide-scope URL of the service, instead of the case-specific URL of an instance. For instance, the *deploy* method in a wiki tool could be the creation of the original empty page.

- The *close* method is used to tell the service that no more activity should be allowed in the instance. Following the example of the wiki, *close* could mean that the instantiated page should be no longer available.

- *set-values* is used to send content or configuration parameters to the service. Different services will support different data types, so the *set-values* element must be accompanied by the mime type that corresponds to data being sent. For example, the wiki example could support the *text/plain* type and use it to set the initial data of the wiki

page.

- *modify-permissions* is used change the users' rights during the enactment. The allowed changes are among those defined in the *permissions* element.

In IMS LD, course participants with different roles are assigned with different tasks, even if the task must be performed within the same tool. It is therefore required the capacity of the service to allow the assignment of different rights to the different users. In GSI, the element that tests the potential granularity of authorization is called *permissions*. The structure of permissions (depicted in Figure 5.3) is explained as follows: the goal is to determine if it is possible for the *holder* to develop a given *action* over certain *data*, where the *holder* is a group[2], the *action* can be *admin*, *write* or *read*, and *data* is determined by its type (*contribution* or *context*) and its *owner* (a GSI group).

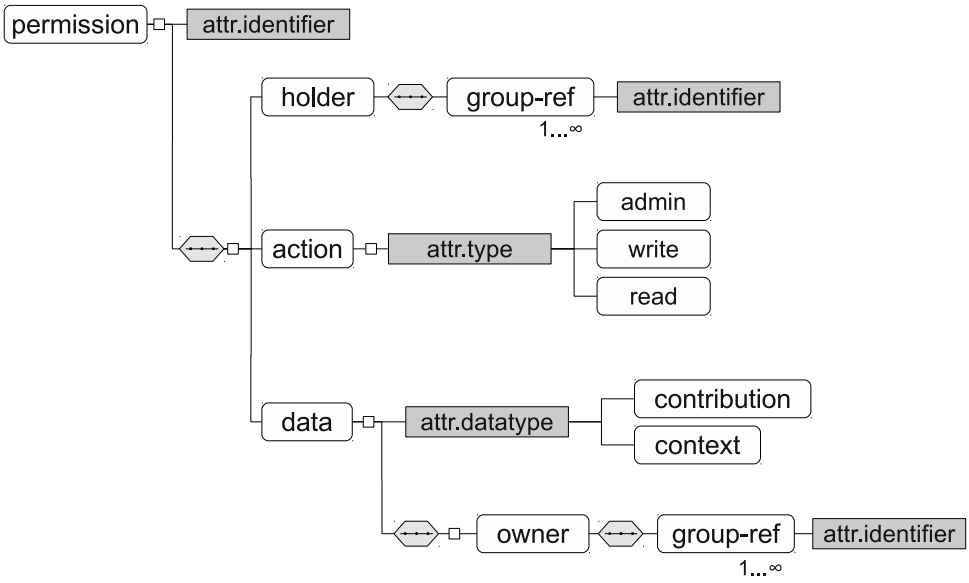


Figure 5.3: GSI permissions definition.

For instance, if the course author decides that the students can only write on the wiki instance of their own team, but they can read all other instances. Then, the permissions will be expressed as in Listing 5.5. The *contribution* and *context* types are respectively used to refer data which belongs to the user and data which is related to the user but not owned by him/her.

The actions are defined incrementally so that *admin* implies *write* and *write* implies *read*. The interpretation of these verbs is case-dependant. That is, these terms have an abstract meaning, while the actual rights related to them may vary on different services. These verbs are capable of capture most of the privileges that can be assigned to a user. The case-sensitivity of the vocabulary is also applicable to the types of *data*: *context* and *contribution*. The former refers to the data that is related to the user, but has not been generated by him or her, the latter is the data directly generated by the user. Again, these two nouns define most of data types.

[2]A group in GSI corresponds to a role in IMS LD

```
<permission>
    <holder>
        <group-ref ref="group-students"/>
    </holder>
    <action type="write"/>
    <data datatype="contribution">
        <owner>
            <group-ref ref="self"/>
        </owner>
    </data>
</permission>
<permission>
    <holder>
        <group-ref ref="group-students"/>
    </holder>
    <action type="read"/>
    <data datatype="contribution">
        <owner>
            <group-ref ref="all"/>
        </owner>
    </data>
</permission>
```

Listing 5.5: "Example of GSI permissions"

The simplicity of the vocabulary that describes *permissions* has been chosen due to the requisite of keeping the complexity of course authoring into reasonable margins. Despite its simplicity, the combination of the three verbs and the two nouns is able to define a wide range of situations, while the actual requirements of services are usually much simpler than that. As stated in Section 5.2.6, the service adapters' documentation is in charge of explaining the concrete mapping among these terms and their actual meaning in the service.

constraints While *groups* define who will use the service and the *tool* states what features should the service accomplish, the *constraints* element is intended to define how and when the service will be used. The possible *constraints* (shown in Listing 5.6) are *life-span*, the *triggers* and *multiplicity*.

The *life-span* defines the duration of the service availability, expressed according to ISO 8601 [192], where the format of duration is PnYnMnDTnHnMnS. For example, "P2Y6M5DT6H30M10S" represents a duration of two years, six months, five days, six hours, thirty minutes, and ten seconds. Shorter durations are expected for learning courses, so PT30M represents thirty minutes and P2D represents two days. If the *deploy* method is used, the time slot starts when the service is deployed. Otherwise, the time slot starts when the Unit of Learning starts being enacted.

When the course is enacted, the IMS LD player reacts to some events such as the completion of an activity or the modification of a property. The *triggers* element defines the events whose reaction is the execution of a GSI command. This element also defines which is the command to execute and the parameters to use. A GSI command can be triggered by four different events: the deployment of the service (*startup-action*), the completion of a IMS LD activity (*on-complete-action*), the accomplishment of a IMS LD condition(*on-condition-action*) and the closing of the service (*finish-action*).

Note that the *on-complete-action* contains a reference to a IMS LD element. That is, this reference links to an element defined out of the *genericService* and thus has to be taken

```
<xs:element name="constraints">
    <xs:complexType>
        <xs:sequence>
            <xs:element maxOccurs="1" minOccurs="0" ref="life-span"/>
            <xs:element ref="triggers"/>
            <xs:element ref="multiplicity"/>
        </xs:sequence>
    </xs:complexType>
</xs:element>
...
<xs:element name="triggers">
    <xs:complexType>
        <xs:all>
            <xs:element ref="startup-action" minOccurs="0"/>
            <xs:element ref="finish-action" minOccurs="0"/>
            <xs:element ref="on-complete-action" minOccurs="0"/>
            <xs:element ref="on-condition-action" minOccurs="0"/>
        </xs:all>
    </xs:complexType>
</xs:element>
...
<xs:element name="multiplicity">
    <xs:complexType>
        <xs:attributeGroup name="attr.multiplicity-type">
            <xs:attribute name="type" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:token">
                        <xs:enumeration value="one-per-group"/>
                        <xs:enumeration value="one-per-user"/>
                        <xs:enumeration value="one-for-all"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:attributeGroup>
    </xs:complexType>
</xs:element>
```

Listing 5.6: "GSI constraints definition"

into account if the service definition is going to be reused[3].

The *multiplicity* refers to the number of required service instances. There are three possibilities: *one-for-all*, if all the course participants will share the same instance; *one-per-group*, if the instance is shared by members of the same role; and *one-per-user*, if the instances have individual scope.

external-value

In IMS LD, the adaptation of content is based on the so called properties, which are tested by conditions and may result on the show or hide action of a piece of content. Thus, to base the adaptation in certain information, it is first required to store this information into a property. The *external-value* element is used to retrieve such information from the external service.

The values retrieved from the external service vary from one tool to another, so that it is not possible to point to the exact data. Instead, GSI offers a simple vocabulary that, in conjunction with case specific adapters (see Section 5.2.6), allow the retrieval of specific data with a generic nomenclature.

[3]A similar thing happens with GSI groups and IMS LD roles

```
1  <xs:element name="external-value">
2      <xs:complexType ref="externalValueContentType"/>
3  </xs:element>
4  ...
5  <xs:complexType name="externalValueContentType">
6      <xs:choice>
7          <xs:element ref="custom-value"/>
8          <xs:element ref="contribution-value"/>
9          <xs:element ref="context-value"/>
10     </xs:choice>
11 </xs:complexType>
12 ...
13 <xs:element name="context-value">
14     <xs:complexType>
15         <xs:attributeGroup name="attr.contribContext.req">
16             <xs:attribute name="serviceref" type="xs:IDREF" use="required"/>
17             <xs:attribute name="owner" type="xs:IDREF" use="required"/>
18             <xs:attribute name="position" type="xs:nonNegativeInteger"
                   use="required"/>
19         </xs:attributeGroup>
20     </xs:complexType>
21 </xs:element>
```

Listing 5.7: "GSI external-value definition"

The *external-value* element, whose content is shown in Listing 5.7, can be placed inside the IMS LD *property-value* element. The course author assumes that the service will offer the information as an array of untyped data, and that the different values are accessed through the *position* parameter.

The service may offer several arrays, depending on the type of value requested: *contribution*, *context* and *custom* are available types. The first two, contribution and context, have the same definition given for the data types in permissions. As well as in permissions definition, they also can point to the data of a certain owner.

The *custom* type corresponds to data that can be manually labelled in the external service with a given tag, so that "custom data" means "data labelled with the proper tag". The *custom* value allows marking data during the enactment, in case the needed information was not available during the authoring.

One of the design principles of the framework is to keep at the minimum the inclusion of IMS LD elements in GSI ones and vice versa. The model detailed in this section shows three linkage elements, explained in Table 5.1.

5.2.3 Service selection

Service descriptions, given by the vocabulary presented in Section 5.2.2, are stored in the manifest and packaged with the rest of the Unit of Learning. Services are described in a generic manner: instead of imposing a specific tool, the vocabulary describes the functionality that is expected. Course authors are allowed to complete this generic description with a recommendation of a specific tool, but the availability of the recommended service during deployment is not guaranteed.

Based on the service description included in the UoL, the runtime engine must select a service that matches the given requirements. The *keywords* list is used to delimit the set of services candidates to be used. The next step is to select the most apropriated service and, depending on the capabilities of the runtime engine, this selection can be automated or may require manual intervention.

Element	Relationship	Description
genericService	imsld environment contains genericService	The service is not the activity, is a tool that supports the activity.
group	a group refers to a single role	The GSI description always refers to the group, instead of the role.
on-complete-action	on-complete-action refers to an activity, structure or act	GSI triggers can be executed when the referenced elements is completed.
external-value	property-value contains an external-value	The value of a property is obtained from the external service.

Table 5.1: Link points between GSI and IMS LD elements.

In the developed prototype, the adapter selection is performed when the Unit of Learning is being imported: the platform detects the presence of a *genericService* element and redirects the teacher (or the person who is uploading the course) to the service configuration page (Figure 5.4). The teacher should select the adapter that best matches the expected functionality among the available ones (Figure 5.5).

A service adapter is a piece of software that performs the information exchange between the external service and the GSI layer, which communicates with the IMS LD workflow engine. As shown in Figure 5.6, the platform can have several installed adapters that are independent among themselves. In the deployment phase, the IMS LD engine needs to instantiate the service, so it is first needed to select the proper adapter.

When the service adapter has been chosen, the selection is used for all course instances of the same UoL. That is, the service selection process is only required when the course package is imported, regardless the number of instances that this UoL will generate. There could be cases where it is required the use of different service adapters in different course instances. In those cases, the same course package should be imported as many times as different services are needed.

Immediately after the selection of the adapter, and before the service is instantiated, the next step of the course life-cycle is prepared and the *act zero* is created. The *act zero*, explained in detail in Section 5.2.4, is the method used by GSI to let course participants to take part in the configuration process, in case they are required.

More than one service can be included in the same Unit of Learning. In this case, the course is not ready to be enacted until all the *genericService* have been properly related to their corresponding adapters.

The GSI framework does not guarantee the selected service adapter to fit the requirements imposed by the course. The deployment platform is responsible of making a good selection. In the developed prototype, the responsibility relies in the person who imports the course, usually the teacher. In all the experiences developed for the purpose of this dissertation, the service has been effectively selected with the expected functionality. However, it could happen that the chosen service does not match the expected functionality. In this case, the activity that depends on the external service will not be successfully performed and the pedagogical sense of the learning flow will be compromised. The development of methods to guarantee the feasibility of the selected adapters is therefore of the greatest

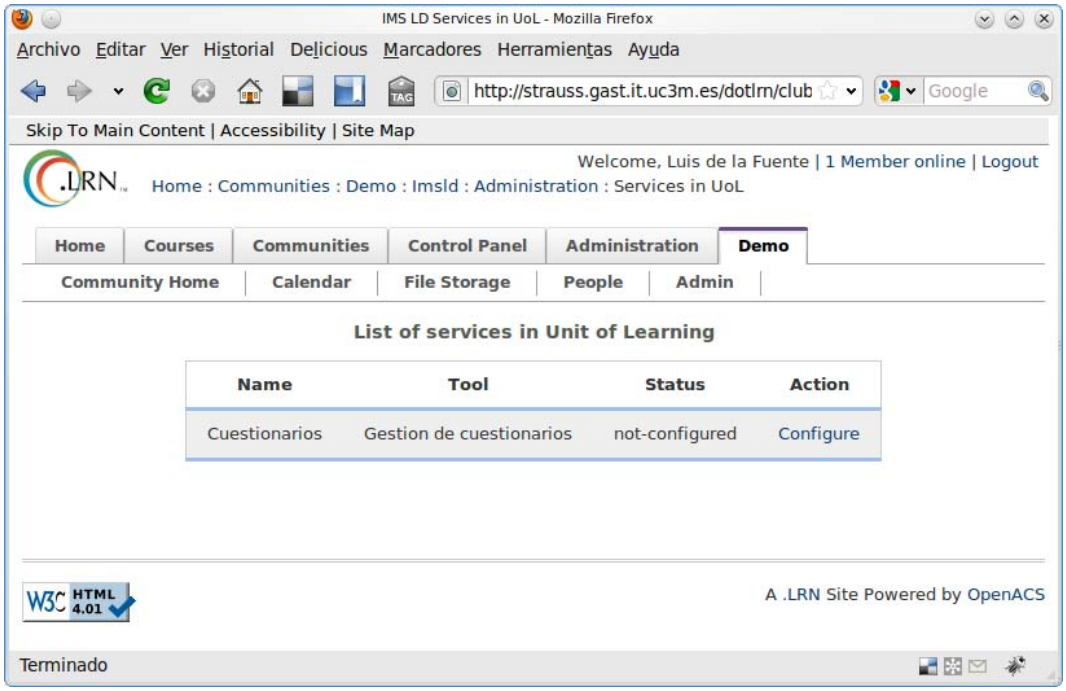


Figure 5.4: Screenshot of the service configuration page.

importance, and should be discussed in future works (see Section 8.2).

5.2.4 Service configuration

In the course life-cycle, the deployment phase is devoted to instantiate all services and resources in order to allow their interaction with course participants. This interaction occurs in the enactment, which is said to be started when the course participants first access to the course.

However, the configuration of third party tools usually requires manual intervention of all course participants (see Section 5.3), so they should take part in the course deployment. This intervention of course participants during service configuration presents a problem to platform administrators: the configuration interface is usually only available for administrators, while course participants can only interact with the course instance when all resources and services have been allocated. The goal then is to allow non-privileged users to participate in course configuration without letting them access the administrator interface nor creating a new specific user interface, which would decrease the tool usability.

The adopted solution in GSI was to include the so called *zero act*, which is a set of activities that are presented to the user as if they were part of the course activities, but they do not belong to the original UoL. If required, a GSI service can include one or more activities in the *zero act*. These activities are similar to other regular ones, but they are not part of the pedagogical flow, they are used to configure the required external services and have been included in the activity tree by service adapters. Thus, the actual course begins when the zero act finishes.

To introduce the *zero act*, the GSI framework redefines the course life-cycle by including a new phase called *pre-enactment* that includes the *zero act*. The *pre-enactment* is defined as the phase that starts when the course instance is created but services have not been instan-

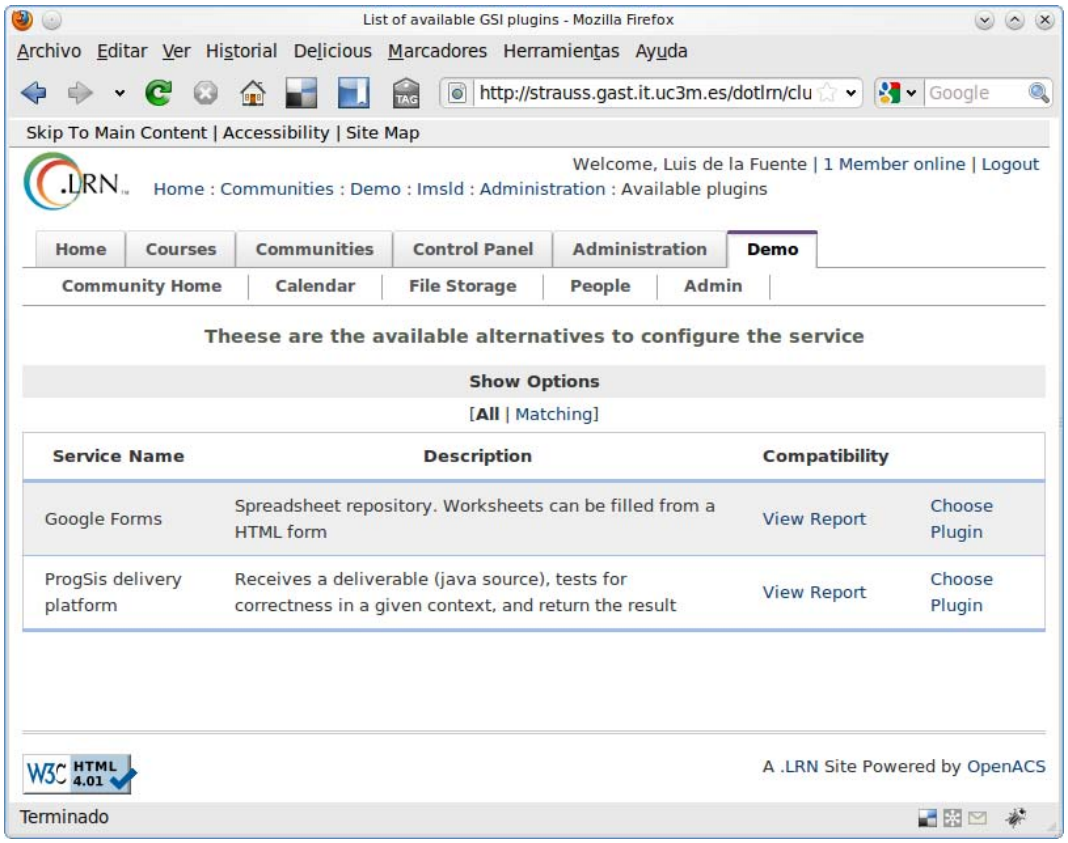


Figure 5.5: Screenshot of the available adapters list configuration page.

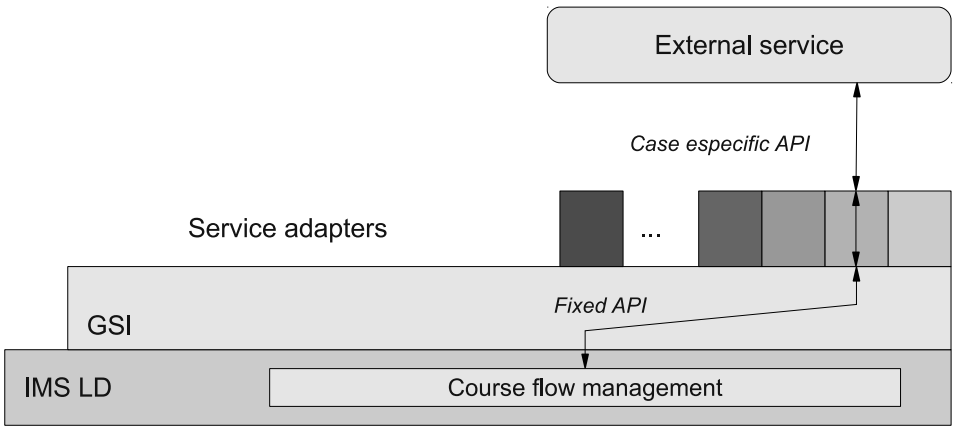


Figure 5.6: Layered architecture proposed by GSI.

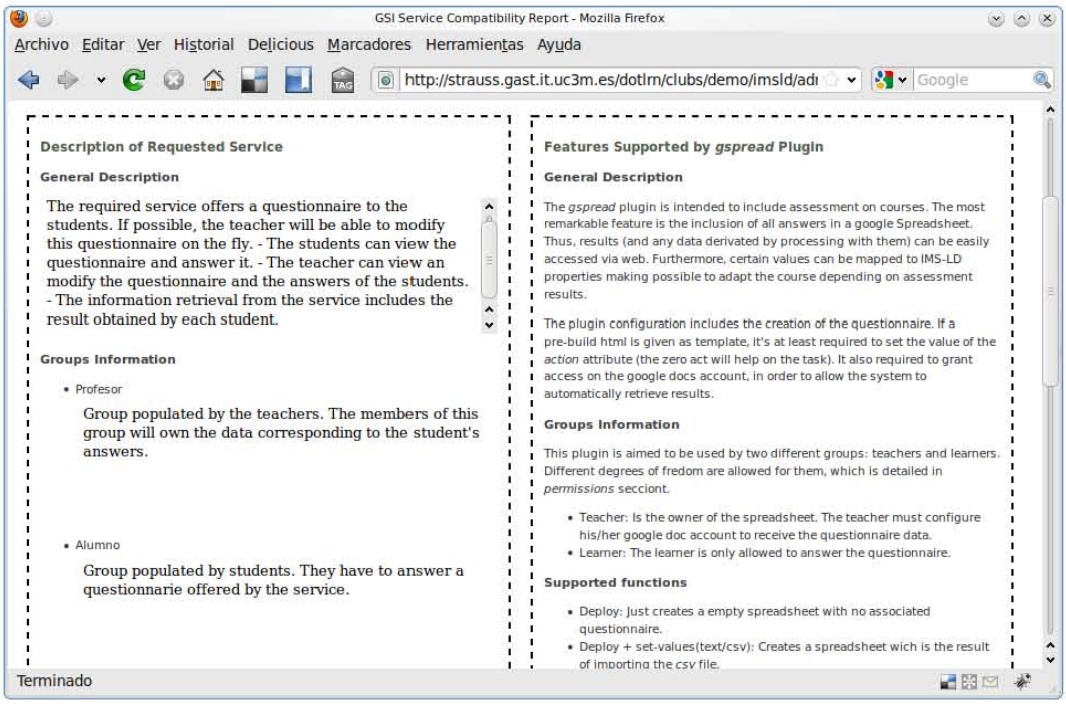


Figure 5.7: Adapter compatibility report presented to the teacher.

tiated, and finishes when course participants have performed their requested configuration tasks and the service instances have been created.

The *zero act* is created and inserted in the course by the service adapters. In other words, *zero acts* are case-specific and the configuration activity depends on the adapter to be configured. GSI imposes the following rules for the *zero act* creation:

- Only one *zero act* is allowed in a course. If the Unit of Learning contains two services or more, all of the adapters will include their configuration activities in the same act.

- The activities to be performed in the *zero act* must be included in a structure of the sequence type. If the Unit of Learning contains two services or more, there will be one structure per each adapter to configure.

- The activities of the *zero act* must be linearly delivered to the user. Neither loops nor adaptation are allowed in this activities. All the activities in the *zero act* are manually finished by the user (in the same manner as regular activities whose completion rule is *user-choice*.

- Not all the roles are required to participate in the activities of the *zero act*. The service adapter must indicate which roles take part in the activities, which depends on the *permissions* established in the *genericService* definition of the course.

- The activities in the *zero act* are defined by dynamic Web pages (typically php or tcl scripts), which are provided by the adapter's software. It is allowed for activities to include an environment with one or more learning objects, which must also point to dynamic Web pages provided by the adapter's software.

- The service adapter must chose a "leader role", whose *zero act* completion triggers the finalization of the act for all roles. This feature prevents the actual course to be blocked if one participant does not perform his/her corresponding configuration activities.

There could happen that a given service do not require any activity in the *zero act*. In this case, the service is instantiated when the course instance is created and the *pre-enactment* phase does not appear.

There are two types of administrative tasks that may occur in the *zero act*. They are authentication and manual configuration tasks. It follows an explanation of these two activity types:

Authentication

During the enactment phase, as it is explained in Section 5.2.5, the IMS LD platform request information from the external tool and use it to assign property values. This information is susceptible to be owned by course participants (e.g. the number of entries inserted by each user in a wiki page) so the information retrieval requires the corresponding permission.

The information needed to allow the IMS LD platform to access into the external service is owned by the user, and it is completely private. Course participants should not reveal their credentials, so it is not possible to automate the authentication and authorization process. The *zero act* includes the activities where the IMS LD platform is authorized by the user to access certain information in the external service. A discussion of the possible solutions is given in Section 5.3.

Manual settings

Service adapters offer a functionality that depends on the external service to integrate. The function of the adapter is to translate the information retrieved from the external service into something understandable by GSI. That is, an array of property values. To perform this task, service adapters use the method offered by the external service which is, in most cases, a proprietary API.

There are some cases in which some kind of action or information cannot be performed or retrieved through the proprietary API and it is only available via the human interface. In those cases, manual intervention can be required from course participants. For example, the API used in the adapter presented in Section 5.4.1 allows uploading a document to the external service, but does not allow creating a form based on this document, so this task needs to be done by hand.

The *zero act* contains the activities where course participants have to develop such type of configuration task. Is the developer of the service adapter who decides if it is worth to insert a manual configuration activity or if the piece of information or functionality should be discarded.

5.2.5 Behavior during enactment

A GSI service is included in the course as part of the *environment* element described in the IMS LD specification. Therefore, the tool is not an activity by itself, but a mean to perform the actions specified in the *activity description*. The service is shown to the user as an additional link placed in the corresponding *environment*. During the enactment phase, when course participants perform their activities, the IMS LD engine interacts with the external service in two different ways (see Figure 5.8).

The first type of interaction is programmatically conducted. IMS Learning Design defines when conditions must be evaluated: at the beginning of a course, when a property value has changed, or when an activity is finished. We refer to these moments as *events*. If the evaluation of a condition is true the course reacts by changing the value of a property, changing the visibility of any activity, or by completing an activity. The GSI model extends this interaction as follows:

- The course may react to an event by sending a command to the external service. Thus, the configuration of a service can change depending on course events. The allowed commands (detailed in Section 5.2.2) are *deploy*, *close*, *set-values* and *modify-permissions*.
- IMS LD properties can take their value from the external service, instead of local information. The information is retrieved by the *external-value* mechanism that can be used whenever a property value is refreshed.

The second type of interaction is directly derived from the users' activity. The course material contains a set of hyperlinks to local and external resources. The course participants use these links to access the service and perform the requested activities. Their actions are supervised so that they can be processed and values assigned to properties. Additionally, the regular interaction of students with local resources also produces events that are programmatically processed by the engine.

When the GSI compliant platform interacts with the third-party tool, the data exchange that takes place has certain considerations with respect to the user privacy. An in-depth discussion of this aspect can be found in Section 5.3.

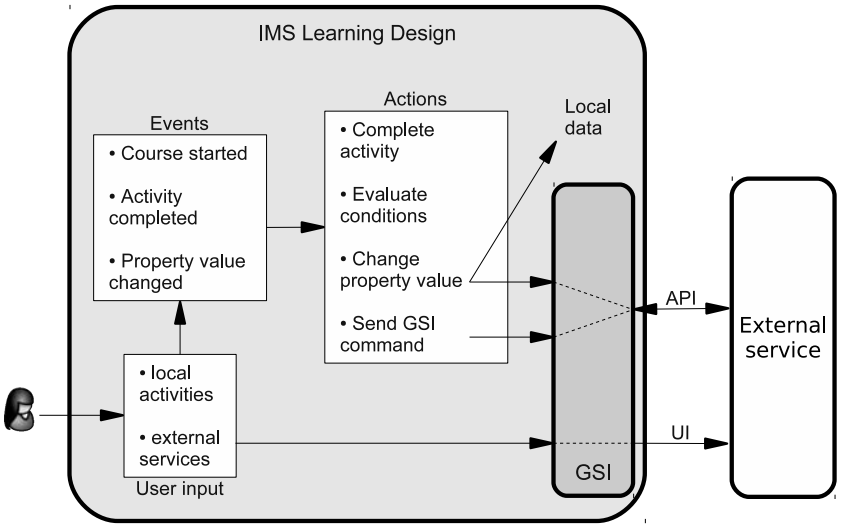


Figure 5.8: Interactions between the IMS LD/GSI engine and the external service.

5.2.6 Translation of the generic generic vocabulary to a case specific behaviour

In GSI, service descriptions are given in a generic manner. That is, the vocabulary is composed by abstract terms whose meaning vary depending on the actual context where

they are applied. A service description needs to be tied to a specific service in order to be completely meaningful. Some of the GSI vocabulary elements do not depend on the actual service. For example, the *triggers* define when *functions* will be executed, regardless of the actual behaviour of the triggered function. There are others whose meaning completely depends on the selected service. These case-dependant elements are enumerated in Table 5.2. The table is supported by the example of a service that provides wiki functionality. More examples of adapters are given in Section 5.4.

Element	**Abstract meaning**	**Example of case-specific meaning: a wiki document**
functions		
deploy	Allow the creation of instances	Each document is an instance. To create an instance is to upload a new document.
close	The availability of instances can be turned off	The document cannot be modified anymore, but can still be read.
mime-type	Data should be sent with the given format	text/plain data is inserted as document content. All other formats are discarded.
permissions		
write	Users can modify the data	Is only considered when used with *contribution* type. Users can edit the text in the wiki document.
admin	Users can modify and manage the data	Is only considered when used with *context* type. Users can edit the tool settings.
contribution	Data that has been included by the user	The user's text.
context	Data that is part of the user workspace	The tool settings: aspect, max. number of simultaneous users, merge policies, save history(y/n), etc.
external-value		
contribution	Data that has been included by the user	In this order: document's text (text/plain); num. of user's contributions; % of user's contribution
context	Data that is part of the user workspace	In this order: num. of contributions; num. of contributors
custom	Data labelled with a given tag	Do not used in this adapter.

Table 5.2: Example GSI elements meaning in the context of the adapter.

The selection of the adapter is performed by the IMS LD platform during the course deployment, as described in Section 5.2.3. The accuracy of the choice will depend on how detailed is the documentation of the service adapter, because the election is performed after a comparison among the *genericService* requirements and the adapters documentation.

The adapter development process is based on the use case that the service will support. That is, a service adapter is created to use the service in a certain manner. This development method implies that:

- Different services are supported by different adapters, even if the external services share a common proprietary API. Thus, the meaning of GSI elements differs from one adapter to another.
- The documentation should include a human understandable explanation of the use case supported by the adapter.
- More than a single use case can be related with the same service. In this case, the adapter developer decides how to face the situation: a unique adapter can provide support to all the use cases, with the consequent increase of usage complexity; or different adapters can support the different use cases, thus increasing the complexity of service selection.

Despite here is no restriction on the development of an adapter that supports several use cases, the design of the framework, development of the prototype and implementation of existing service adapters have been focused in the case where adapters support a single use case. The recommendation for adapters' development is to follow this latter approach.

5.3 Identity and authentication issues

The integration of third-party services in IMS LD described learning flows allow the inclusion of Web 2.0 tools (and therefore Web 2.0 based pedagogical models) into *e-learning* material. By accessing external tools, course participants can perform learning activities with domain specific tools. Furthermore, if data about the user interaction is obtained from the third-party service, the following material in the learning scenario can be adapted.

Programmatic access to external services is usually implemented using proprietary Application Programming Interface (API) offered by the third-party service provided. This type of access does not require user's intervention and does not generate any user session. However, user's data retrieval may be affected by privacy policies restricting the access to data unless the explicit approval by the user is obtained. This fact poses a severe restriction on the described scenario, which can be summarized in the following question: How can the service act on behalf of course participants without affecting their privacy and with no disruption on the learning flow?

This section analyzes the problem of external user management. The proposed solution decomposes the problem into smaller parts where a simpler solution can be applied. The analysis of the problem and the proposed solutions are part of a publication developed within the context of this dissertation [193].

5.3.1 Correspondence of identities

If a Unit of Learning needs external service's data to adapt its appearance or content, the IMS LD server must pragmatically retrieve and store this data. Furthermore, there are cases in which the retrieved data is not related to a given user, but to a more generic environment. For example, the number of times a given resource has been used, or the resource with the

highest ranking according to participant's votes. The opposite case appears when the data belongs to a single user, for instance, the last comment posted from the user in the external service.

The latter presents a challenge to the IMS LD server because the user already has a unique identifier in the local server (e.g. john@LDserver.com). But, assuming he has already been registered in the external service, which is the identifier that corresponds to the same person? As shown in Figure 5.9, the local server needs to find the correspondence between these two identities, so that it is able to retrieve the remote user data and relate it to the proper local user. It may happen that the same person used the same email address to register in both servers. However, it is not necessarily the case, so a procedure to find correspondence of identities needs to be defined.

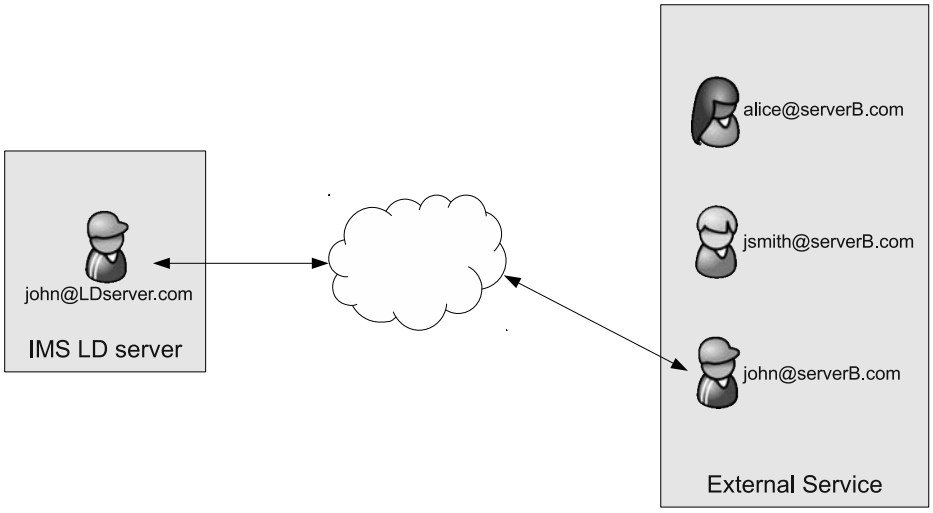


Figure 5.9: A case of use that describes the identities correspondence problem.

Again, the ideal solution where the entire process is automatic requires metadata that associates the accounts in both servers. However, this is not the common case. It follows a discussion of the possible procedures to solve the problem.

The straightforward approach is to prompt the user for her username in the external service. Despite its simplicity, this solution presents some drawbacks:

- Each service included in the UoL introduces one additional activity in the course flow (establish this correspondence). This addition of activities may slow down the course itself, and could result in a negative effect from the point of view of the cognitive load [194].
- Course cannot start until all participants have indicated their external username. It means that if a user does not perform the activity, all the scheduled activities are delayed. Course flow could include alternatives to avoid the problem, but its complexity increases as the number of students grows and hinders the authoring phase.

OpenID is an authentication system that allows a unique, distributed identity to be used on any compliant server [195]. A person who has an OpenID account can use it as registration information in different services, instead of having several usernames and passwords. This system can be used to find correspondence between users in the scenario: if both the IMS

LD platform and the external service provide authentication with OpenID, it is guaranteed that the same identifier corresponds to the same person.

However, this solution also presents some drawbacks, namely:

- There is no restriction in the number of OpenID accounts held by the same person. It is possible for the same person to have different identifiers in different servers. A manual step to confirm correspondence is still needed but it is much less restrictive than in the previous case, because no user action can be considered as the acceptance of the correspondence and does not block the course flow.
- The catalogue of Web tools that have adopted OpenID restricts the number of available services to be included in a UoL.

The third solution, based on the use of OAuth [196], provides a way to find this correspondence and authorize the access user data both at the same time. This solution is discussed in Section 5.3.2.

5.3.2 Authentication for data retrieval

A service that offers data through the public API must ensure that the other party owns the corresponding privileges. Typically, the owner of the data (the student) must actively grant access to her data.

The IMS LD server could prompt students for external service's login and, additionally, password. The privacy problem is clear here: there is no reason for users to reveal their password to anybody. In the hypothetical case in which the user trust the IMS LD server and facilitates her data, security problems such as man-in-the-middle attacks [197] rely on eavesdropping an insecure channel and if this takes place during the authentication phase, the attacker is able to gain access to the user credentials. As a result, storing the user credentials for the external site in the IMS LD server is not recommended either because it compromises these credentials.

The most suitable solution is to use the authentication protocol known as OAuth [196]. The use of such protocol requires the following steps:

- The requester agent, in this case the IMS LD server, initiates the OAuth negotiation with the external service. As a response, the external service provides a *request token* to the IMS LD server. The external service will also provide a token secret to be used together with the request token.
- The IMS LD server then redirects the user's browser to a previously set URL, hosted by the third-party service. If the user has not started a session in the service yet, there will be a redirection to a sign-in form; and after the user introduces the proper credentials, a request to allow access to the IMS LD server is presented.
- When the user agrees to allow the IMS LD server to access the information stored in the external service, the IMS LD server receives an *access token*, which differs from the first one in that it is an authenticated token, and it can be used as proof of authentication to retrieve data from the external server.
- In the next interactions between the IMS LD server and the external service, the IMS LD server uses both the access and the secret tokens. The external service is then able to validate if the tokens are correct and if they belong to the owner of the requested data. If these conditions are met, then the external service serves the requested information.

A graphical representation of these steps is shown in Figure 5.10. In this example, the IMS LD server wants to retrieve information belonging to user John, which is provided by an external service called *generic* (G). In the worst case, John would have to provide the credentials he uses in service G to the IMS LD server; instead of this, service G offers an API with authentication performed through OAuth. Therefore, John allows service G to provide his information to the IMS LD server through the authentication OAuth token. This token, stored and used by the IMS LD server, is implicitly related to John's identity on the third-party service. That is, the correspondence between identities has also been established, implicitly solving the problem stated in Section 5.3.1.

One of the disadvantages of OAuth is the lack of granularity for privileges within its permission system, since the only capability provided to the user is either to grant or not all privileges to the IMS LD server. This low granularity implies that even if the IMS LD server needs only read privilege to a small portion of information, it will be granted full permissions, depending on the actions provided through the API by the external service.

Another drawback of OAuth is the constant lifetime of the authentication token. The lack of flexibility regarding the lifetime of the token affects in a special way those courses with duration longer than the period of time in which the token will be available. In this scenario, the learners would see their UoL interrupted at some point, since the communication between the IMS LD server and the external service would need the token to be updated in order to continue with the validation.

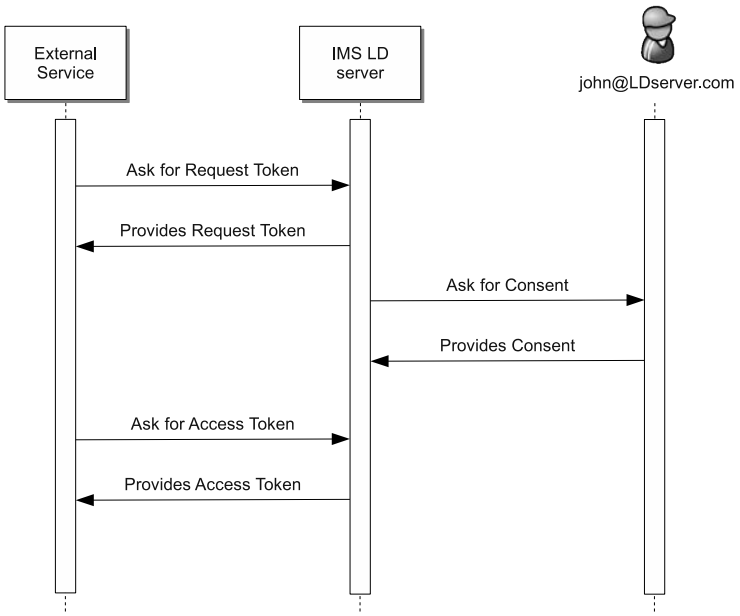


Figure 5.10: Exchange of messages between IMS LD server and external service to authenticate user John.

5.3.3 Automatic user creation

As discussed in Subsection 5.3.1, there is a need to find a correspondence of user accounts between different services. This correspondence issue translates into a more specific problem,

which is the automatic creation of a user account in the external service.

The previously presented problems rely on the assumption that the user has previously created an account within the external service. Considering the amount of possible learners involved in the UoL and the large number of external services available, it is very unlikely that every learner will have an account in all the used external services.

Current anti-spam technologies, such as "captchas" [198], precludes the possibility of creating user accounts automatically in the external service. Thus, a different approach is required.

In a scenario where all services are controlled by the same authentication entity, such as corporative platforms, this issue is not present. When using technologies such as LDAP or ActiveDirectory, one credential allows access to all the services. However, this centralization is what makes this solution less feasible for open and decentralized scenarios.

Consequently, a courseflow paradigm opened to the Web 2.0 is assumed to require user's intervention during the creation of their accounts. The goal is to minimize time consumption during this process. The most feasible solution is the use of OpenID to delegate the management of user accounts to a third-party service. An identity managed by an OpenID provider is complemented by user's data, so that they can be used to create the desired user account.

If both the IMS LD and the third-party service provide OpenID support, the OpenID identifier can be directly used in the creation of the account. The third-party service asks the OpenID provider for additional user's data. The process finishes when the user herself accepts the creation of the new account.

However, this solution presents difficulties when used in an educational environment:

- Although the external server uses OpenID to authenticate users, it may require an additional registration to match the OpenID with an account. As a consequence, although the process for signing up is eased by the use of OpenID, the user needs to deviate slightly from the instructional flow described in the UoL in order to register within the external service.

- Some services may require more information than the one provided by the OpenID provider, it is usually not possible to provide this information in an automatic way through an API, therefore this has to be taken into account when choosing what external service to use with the IMS LD server. Passing the fulfilment of this critical information to the learner might bring problems during the execution of the UoL and to the flow of the whole course in general.

5.4 *Generic Service Integration* in GRAIL

The support for the GSI model has been implemented in the GRAIL IMS LD player, which has been used to deploy the experiences described in Chapters 6 and 7. According to the rest of the underlying platform, the GSI support was developed with the *tcl* programming language and *postgresql* was the database in use.

One of the GSI design principles, explained in Section 5.2.2, was to minimize the situations in which GSI elements include or refer IMS LD elements. The same philosophy was followed in the development of GSI support: the impact of GSI specific code on the already existing IMS LD code has been reduced to the minimum. The only points where existing code has been substituted by GSI specific instructions are:

- The creation of the application datamodel, modified with the inclusion of GSI tables.

- The manifest parsing, modified with the capability of managing *genericService* elements.
- The instantiation of a course, modified with the instantiation of the GSI service.
- The value assignment to a property, modified with support for the *external-value* type.
- The execution of a GSI command when an activity is completed.
- The rendering of the user interface, including the access to the service instance.

The rest of the implemented code is located in different files. Thus, GSI support is developed in the same package but the code is clearly differentiated from the rest of the GRAIL code. This distinction emphasizes the relevance of a modular source code for its maintenance or improvement.

A similar separation-of-concerns approach was taken in the development of the tool datamodel, which is a practical implementation of the abstract model depicted in Section 5.2.2. Figure 5.11 depicts the relational datamodel used to store GSI objects in the postgresql database.

In the implementation provided, adapters are not part of the GRAIL package. According to the architecture depicted in Figure 5.6, service adapters are implemented as independent software that can be added to and removed from the GSI module. Service adapters are implemented over the OpenACS toolkit and its usage is meaningful when it happen in the context of GRAIL courses and is mediated by GSI. The architecture is depicted in Figure 5.12, which is constructed over Figure 4.2 from Section 4.2.2. The connection between GSI and the service adapters is performed by an API that must be implemented by all adapters. A design principle of GSI is to keep the development of new adapters as easy as possible. Guidelines for such development are given in Subsection 5.4.3.

5.4.1 Google Forms and Spreadsheets

The first service adapter developed for GSI is the support of Google Forms and Google Spreadsheets. The adapter, called *gSpread*, aims at the provision of assessment functionalities in the context of IMS LD courses.

Google Forms [199] is a Web tool offered by Google that provides an easy method for questionnaires creation and management. Such tool offers an interface that allows the easy creation and edition of Web questionnaires. The created questionnaires can be published so that everyone who know the URL can access and provide an answer (no private publication is allowed). The answers are stored in a spreadsheet hosted by Google Docs [200], owned by the questionnaire creator.

Having this functionality, the case of use supported by the adapter is described as follows:

1. When the course is instantiated, a new Google Spreadsheet is created and owned by the teacher.
2. The teacher is in charge of the questionnaire creation, usually with the help of a local document with the questions. The questionnaire could also be included in the UoL package so that its reuse is simpler.
3. Students can access to the questionnaire and they give their answers. The teacher can access both the questions and the answers, so that he/she monitors if all the students have properly answered. The adapter forces the inclusion of two fields in the students' answer: their name and a security mark (i.e. a fingerprint).

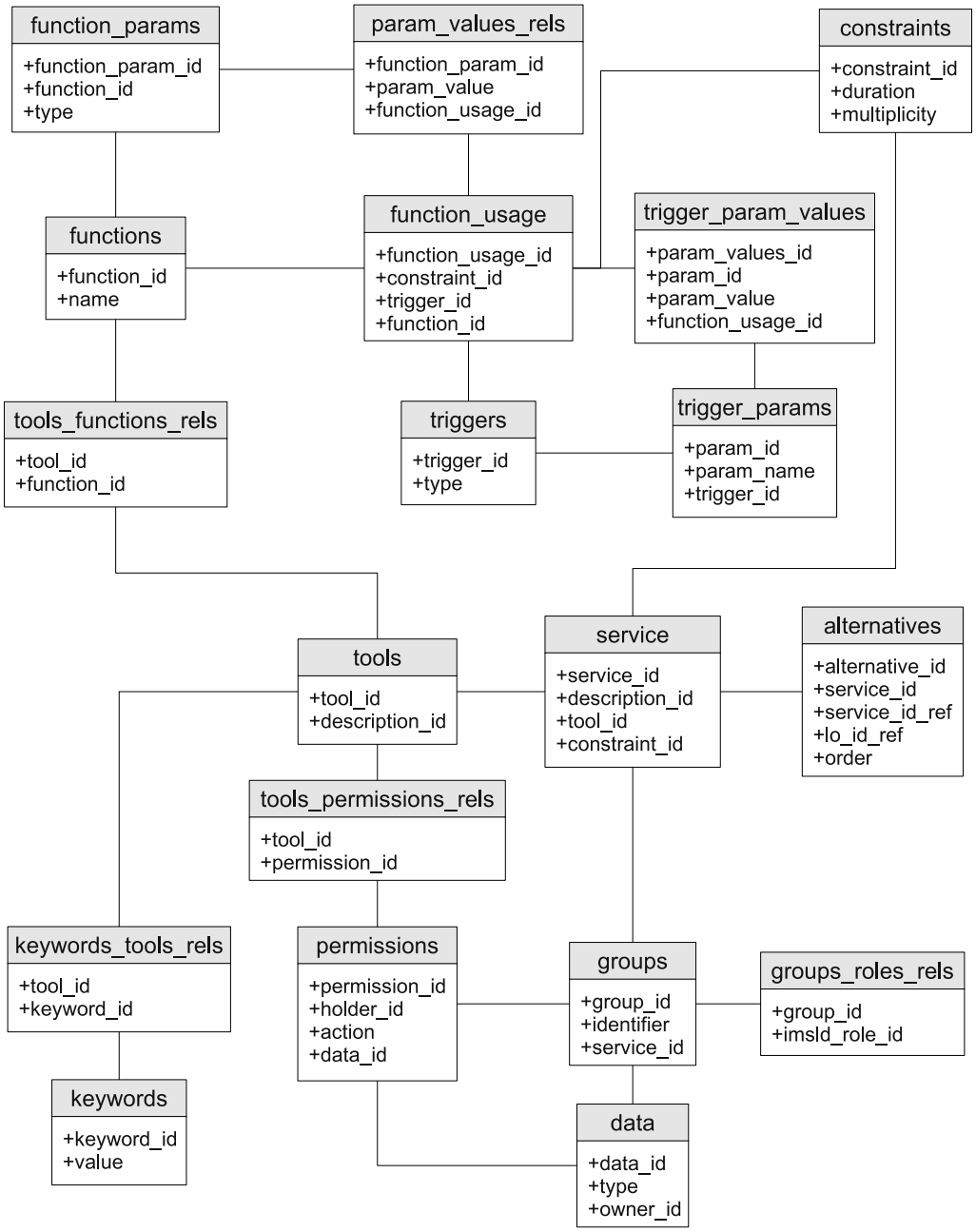


Figure 5.11: Database class diagram of GSI support in GRAIL.

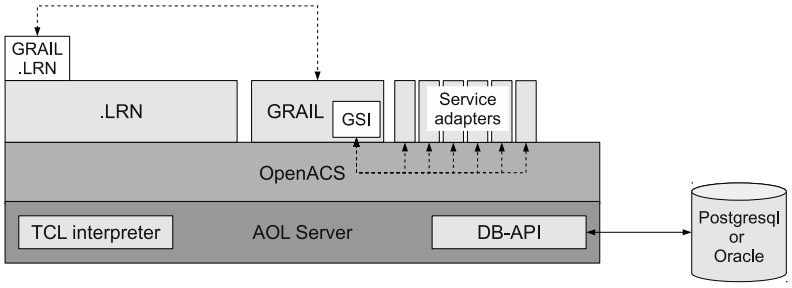


Figure 5.12: Architecture of the GSI support in .LRN.

4. The teacher operates the spreadsheet so that he/she finds a value that represents the score obtained by each student.

5. The activity flow is adapted to the individuals depending on their obtained score.

The *gSpread* adapter supports the mime types *text/csv* and *text/html*. The latter is used to set the questionnaire to be answered when it is packaged within the UoL. The *csv* format is valid to set the fields (that is, values in the columns of the first row) with which the spreadsheet is created. The possible supported combinations are: the provision no parameter, if the teacher must create the questionnaire from scratch; just the *csv* file, if the questionnaire will be created online but the spreadsheet fields are predetermined; or both mime types, when the questionnaire and the spreadsheet fields are provided within the package.

The adapter has been designed to be used with *one-for-all* multiplicity. That is, the same service instance is used by all course participants and only owned by the teacher. In the typical supported case of use, the teacher has *context-admin* permission, which means that he/she can completely manipulate the questionnaire and the students' answers. The students should have *contribution-write* permissions, due to their only allowed action is to answer the questionnaire. Other combinations are allowed, being their behaviour one of the two described possibilities.

The treatment of the *external-value* element ignores the *context* modifier and only considers *contribution* and *custom* as valid modifiers. The *contribution* modifier returns the last row of the user (including the answers to the questionnaire and the values calculated by the teacher). The *custom* modifier is used to return the users' cell whose value starts with the given *custom-tag* (e.g. the cell that starts with "result").

Since the teacher is the only actor that actually uses his/her Google Account, the teacher is the only role that is required to perform the *act zero*. There activities included in this act are:

1. The IMS LD server will create and retrieve data from a teacher's spreadsheet, so the teacher has to explicitly grant access to his/her account. Current implementation of the adapter uses *SubAuth* [201], which is a proprietary protocol pretty similar to *OAuth*.

2. Current Google API allows the creation of new spreadsheets, but do not allow to programmatically create Forms. Therefore, the teacher has to manually create the form and get the resulting URL. Additionally, the teacher decides if the he/she creates the questionnaire from scratch or if he/she is going to use the HTML form provided in the UoL.

The *gSpread* service adapter has been used in the development of the experiences described in Chapters 7.2 and 7.4, respectively published in [202, 182].

5.4.2 Delivery Platform

Delivery Platform is the name received by an application developed in the context of *Systems Programming*, a subject taught in the Telecommunications Engineering degree at the University Carlos III of Madrid. The tool offers a web interface where the users can upload a file with Java source code. Then, the *Delivery Platform* executes some pre-configured test cases on the uploaded code and says if the Java source code passed the tests or not. Each uploaded file is classified by the name of the submmitters, their group, and the identifier of the test case to be executed. These three values are provided by the user when he/she uploads the file. All results are stored and presented in a Web page that requires authentication.

The developed adapter, called *delivery-platform*, provides the opportunity to orchestrate learning activities in which code tests are performed. The case of use supported by the adapter is described as follows:

1. The teacher develops Java tests cases, whose execution will check if the functionality of certain Java file is compliant with the requisites of the activity.
2. The teacher configures the Delivery Platform so that the developed test case is associated with an identifier.
3. Students access the submission formulary that corresponds to the configured test case and upload their solution.
4. If the code success in the testing process, then the next activity of the course will be available. Otherwise, the students have to revise their code and submit it again.

The service adapter allows the execution of the *set-values* method, which has the following meaning: the *text/plain* mime type is used to establish the identifier whose submissions form will be retrieved. When the service is being instantiated, it is also allowed the use of *text/x-java-source* or *application/x-jar* in order to configure the test case that will be used in the service.

The service supports the three allowed multiplicities: if it is set to *one-per-user*, the student will only pass the activity if he/she submits the right code. In *one-per-group* all students in the same group pass the activity if one of the group members uploads a good solution. When the multiplicity is *one-for-all*, just one good solution is enough to let all students to pass the activity. Teachers are supposed to have *admin-context* permissions that let them to submit code, view other results and download all submitted code. The students can hold *contribution-write* permission that lets them to submit code and view their own results or *contribution-read* if they are only able to view results. Other permissions are ignored in the current version of the adapter. The data offered by the service is the result that the student obtained in the last performed test. Thus, the *position* parameter passed to the *external-value* element refers to the test case identifier.

The *delivery platform* service adapter has been used in the development of the experience described in Chapter 7.3 and published in [183].

5.4.3 Guidelines for adapters development

This section has explained the GSI architecture, the relevance of service adapters on the overall model and has described two examples of already implemented services. GSI recognizes the adapters as an essential part of the model. If the creation of new adapters poses

a challenge to developers, then there will be never developed. Thus, the complexity of the creation process should be reduced to the minimum.

This subsection summarizes the sequence of activities required for a successful implementation of new adapters. The text does not attempt to provide an extensive development guide, which is out of scope of this dissertation, but a glance at the whole process.

1. **Define a use case for the service**. Web 2.0 tools provide rich functionalities whose educational use can be done in different manners. In order to simplify its development, the adapter provides support for a single case of use. The first task on the implementation process is the definition of the case of use that will be supported. If the adapter needs to support more than a single case of use, it is at the cost of more complexity on its implementation and its use in courses.

2. **Express the use case in terms of GSI**. The vocabulary offered by GSI allows describing how the service will be used. Thus, it should be clear how *permissions*, *functions*, *mime-types*, *offered data arrays* and *keywords* are combined so that they better describe the supported case of use. The mapping between human description of the case of use and GSI elements provides a guide to developers in the implementation of the source code.

3. **Implement the actual code**. Based on the case of use given at the previous steps, code developers must provide an implementation of the methods given in Appendix A. Since the existing implementation uses *tcl*, this is the language used in the Listing. However, the methods could be easily extrapolated to other programming languages if it is suggested by the underlying platform.

4. **Write documentation**. The last step is to provide a human readable description of the adapter characteristics that will help course managers in the adapter selection process. The human definition of the case of use and its expression with GSI vocabulary should be reused in this step. The documentation should not be a guide for code understanding, it is an explanation of how to use the adapter.

5.5 Discussion of the model

At the beginning of this chapter, in Section 5.2.1, it is stated that "A constraint imposed to the model is not to break the following intrinsic characteristics of IMS LD: *Pedagogical neutrality*, *Reusability*, *Self-containment*, *Collaboration* and *Adaptability*". After having presented in detail the GSI model, it is worth to revise how the imposed constraint has been satisfied. This section discusses how the model supports these characteristics.

Pedagogical neutrality. The GSI framework does not restrict the type of services that can be used, with the only requisite of being available via Web. All services have the same threshold to be used in learning courses: the implementation of the corresponding adapter. Furthermore, the case of use of the adapter could be oriented towards the support of a certain pedagogical method, or it can be neutral. It can be concluded that the use of GSI does not narrow the range of pedagogical models allowed by IMS LD.

Reusability. Lessons learned from the experiences described in Section 6.3 dictates that, in learning courses, the bottleneck of reusability is the time spent in course replication. In other words, if there is a big effort required to enact another replica of the course, the cost of deployment will not worth the potential learning benefits. The

GSI framework contemplates the possibility of automatic service instantiation. As demonstrated in experiences described in Chapter 7, GSI-based course replicas can be quickly instantiated so the bottleneck is overcome if there is a robust automatic service instantiation.

Self-containment A course package is said to be self-contained if all the information required to successfully deploy and enact the learning flow is inside the package. In the case of GSI, the service description is used to select the best service adapter, which will be used in all course replicas in this platform. Thus, all course instances will behave exactly the same, so the self-containment characteristic is also accomplished. In is also possible to import the same course package in different platforms and select different service adapters in each case. This is not a problem if both service adapters satisfy the requisites imposed by the service description and therefore both course replicas can be successfully enacted. However, if there is a bad service selection, it could happen that some of the activities will not be able to be performed, and the support to the learning flow will not be enough. Consequently, the self-containment characteristic depends on the capability of the deployment platform to select the best service in each case.

Adaptability IMS LD support for adaptive content is discussed in [126, 127, 129, 130], where the conclusion that *properties* and *conditions* are the vehicles to adaptation in IMS LD. There exist some types of adaptation and the support of all of them requires the use of properties. Therefore, if property values are set depending on external sources of information, then the course flow adaptation will be able to be based on the use of these external sources. Consequently, the ability to adapt course material depending on the participants' behaviour on the external service is limited to the data that can be retrieved from it. That is, the more data are contained in the data arrays offered by the service adapter, the more fine grained adaptation can be implemented. GSI support of adaptability depends on the case of use provided by the service adapter and the data included in the offered data arrays.

Collaboration As discussed in some documented experiences (see [111, 117, 121] for a sample) and also confirmed by the experiences described in Section 6.3, IMS LD supports the recreation of collaborative learning flows by means of *roles* and/or a proper use of *properties*. The latter has been discussed in the previous point. Roles are supported in GSI, where they are called *groups*. Different groups may receive different permissions in the external service and, depending on the multiplicity, the actions performed by teammates can affect the user's learning flow. Collaborative learning flows with GSI service integration have been successfully deployed in the experiences described in Chapters 7.3 and 7.4. Therefore, it can be said that GSI supports collaboration as far as IMS LD does.

5.6 Conclusions

This chapter has presented the *Generic Service Integration* framework as a proposal that allows IMS LD to integrate external services in learning courses. GSI offers a vocabulary with three main parts: *tool* to define what the characteristics of the external service are; *groups* to define who will use it; and *constraints* to specify how and when the service will be used. External services are included in the IMS LD manifest file as a generic description of requested features, emplaced in an environment of the UoL. This generic description is translated into a concrete service during the course deployment phase. Then, the IMS LD player is able to bidirectionally exchange information with the external service at course

enactment, so the learning material can be adapted depending on students' behaviour on the external tool.

The design of the GSI framework has presented two major challenges. First, the lack of a common protocol that allows the communication with the external tools, which usually offer proprietary APIs to provide their data. The proposed solution is an architecture based on the use of service adapters, where these pieces of software are easy to implement and plug in the system. The second challenge is the management of users' authentication within the external service. The inclusion of the *act zero* allows users to grant access to their data in the third-party tool, so that the IMS LD can programmatically retrieve this data without the need of the users revealing their private credentials.

The GSI has been implemented as part of the GRAIL player, build over the .LRN Learning Management System. Current implementation offers two service adapters (*gSpread* and *delivery-platform*) that have been used in different learning experiences described in the following chapters. The implementation also has allowed obtaining a quick and well-defined adapter development process that allows new services to be easily integrated.

In summary, GSI allow the integration with external services without breaking the intrinsic characteristics of IMS LD: pedagogical neutrality, reusability, self-containment, adaptability and collaborative capability.

Chapter 6

Motivating Experiences

> The illiterates in the future will not be those who do not know how to read and write, but those who do not know how to learn, unlearn, and relearn.
>
> Alvin Toffler

6.1 Introduction

At the time that the work presented in this dissertation started, the state of the art regarding the IMS LD specification did not provide neither mature enough tools nor experiences. Several tools and experiences have been published since then, as stated in Chapter 3. This chapter presents the experiences deployed for the study of the specification, the first phase of the research methodology. In order to better understand these experiences, it is worth to depict the situation at the time this dissertation started (2006).

First, existing tools were still in an immature state: the authoring stage was supported by Reload (conceptually close to the datamodel) [86], LAMS (not fully IMS LD compliant) [92] and Collage (pattern based) [93]. As an example of such immature state, let's say that the first version of Coppercore was released in February 2004, the same year that Reload was. The first version of the GRAIL player was just released. That is, its functionality covered the three levels of the specification but there were no large-scoped tests posed on the tool.

The discussions about the specification were centred on how different pedagogical models were supported, rather than on providing effective authoring tools that hide the underlying model. The earlier IMS LD-related literature theoretically argues the support for different pedagogical models. For example, collaborative learning was said to be supported by the introduction of roles [119], while properties and conditions was considered enough to create adaptive learning material [127]. Regardless the validity of these assertions, few experiences neither confirmed nor refuted the arguments. Almost all of the available UoLs consisted in proofs of concept that, with simple learning flows, showed how different models could be applied. However, among those examples there was an abscense of experiences in real scenarios. For example, *What is Candidas* were used to show the sequencing capabilities and *GeoQuiz* showed adaptive capabilities. All these examples, available at [131], were synthetic learning flows that did not answer the questions that usually emerge when the learning flow of a complete term is being designed.

The described context influenced the methodology used in this dissertation, presented in Chapter 1.3. Thus, the first step was oriented towards the identification of IMS LD limitations in authentic situations. The second phase was the definition of the solutions for the detected limitations, which were validated in the experiences held in the last step of the presented work.

The first phase was focused on the analysis of the specific parts or characteristics of the specification which, in the opinion of the researcher, had not been properly documented in the existing literature. The analysis was case study based where each of the experiences held was focused on a particular element of IMS LD. The goal of such experiences was to identify the limitations, being their solution the matter of later steps of the dissertation.

Each deployed experience was oriented towards the analysis of a particular element of the course life-cycle. The goal was to analyse the authoring, deployment and enactment of IMS LD orchestrated courses. The analysis of the authoring phase allowed determining the limits of the IMS LD expressiveness, that is, what can and cannot be expressed in the UoL. The factors affecting reusability and replicability of UoLs were the matter of analysis of the deployment phase. Finally, the enacted experiences were aimed to reveal how users understand the structure and presentation of scripted course.

This chapter presents the experiences held in the first phase of the methodology. The authoring stage is explored in Section 6.2, were a real learning flow, which was already in use in an engineering course, was expressed within the IMS LD vocabulary for its latter deployment in GRAIL. The enactment analysis corresponds to Section 6.3, where a course involving collaboration in distance scenarios was offered to students of three different universities. A theoretical discussion of a more versatile use of IMS LD is presented in Section 6.4. A proof of concept of the discussed Unit of Learning was implemented to support the presented ideas.

6.2 Design and deployment of a real course

The first step in the understanding of the specification is to recreate the complete course life-cycle for real courses. That is, courses built following the needs of the teaching staff and are currently being taught without any scripted orchestration method. In order to avoid the risk of facing a real scenario without enough maturity, this first experience was a simulation of the actual course. Thus, the course authoring phase was completed according to the already established course conditions, but the deployment phase was a simulation of the real thing. The enactment of the generated UoL was out of the scope of this experience.

This section describes the course modelling and deployment process by capturing an already consolidated course in the University Carlos III of Madrid. A blended-learning paradigm was followed in which classroom sessions were combined with students' individual work and information exchange using on-line tools. Its structure was similar to the rest of courses in the same degree. Therefore, analysing the process of capturing such structure in IMS LD provided valuable insight on the advantages and disadvantages of using such formalism. The course deployment simulation allowed identifying the limitations during this step, which can be divided in two: first, problems related with the workflow engine, GRAIL in this case; second, limitations derived from the intrinsic constraints of the specification. The development and analysis of this experience was published in [82].

6.2.1 Objective of the experience

This experience is presented as the first approach to the course life-cycle of IMS LD. The analysis of the experience is performed in a qualitative manner. That is, the goal is to

identify the problems that arise when IMS LD is in use, rather than focusing on a very detailed analysis of specific issues.

More precisely, the specific goals of the developed experience can be expressed as follows:

- First, test the expressiveness of the specification when dealing with consolidated courses that are already being taught. In particular, the modelled learning flow combines synchronous and asynchronous tasks, which occur in parallel and have different timings.
- Second, capture a methodology that allows expressing regular courses as Units of Learning. It should allow practitioners to translate their learning flows into IMS LD scripts.

6.2.2 Course Authoring and Deployment

The steps followed to capture the structure of the on-going course in a Unit of Learning written in IMS LD were three. First, the existing methodology was expressed in natural language. Second, the design was translated to the IMS LD language, creating a runnable UoL. Finally, the designed course was deployed in the IMS LD compliant platform to check its correctness. It follows a detailed description of each of the stages.

The Learning Layout

The selected course was Computer Architecture: a course part of the degree on Telematics Engineering at the University Carlos III of Madrid. The actual course structure is based on two-hour in-class sessions in which either theory concepts are explained, or a programming assignment requiring at least one week work is discussed.

Despite its on-campus nature, the course flow is based on both synchronous and asynchronous activities. On the synchronous side, regular theory lectures present students with the main concepts. Programming exercises are then proposed in the laboratory sessions. These exercises are not expected to be completed during the lab session and therefore require asynchronous interaction with tutors and peers.

An assignment may be submitted several times with partial answers until the next lab session is held. Students therefore are working on a lab and at the same time taking theory sessions about the next topic in the course. Two additional computer based services are provided to ease the remote work. A Web based forum allows students to post their comments or ask tutors for clarification and a second Web based interface is also provided for automatically produced partial laboratory results.

The previous pattern is repeated until all topics in the course have been covered. As show in Figure 6.1, different topics may overlap thus requiring coordination between content and resources.

6.2.3 Learning Flow in IMS LD

Once the learning flow has been defined, the next step is to translate it into a IMS LD course. Although the language is expected to be able to capture a wide range of pedagogical models, the translation process requires extensive knowledge of the specification as to avoid a reliable translation.

The produced course is to be used within a blended learning approach. More precisely, some resources and activities can be obtained and completed in distance sessions, but others will require the attendance to lectures and laboratories. This methodology combination is widely used because of its benefits for both students and teaching staff.

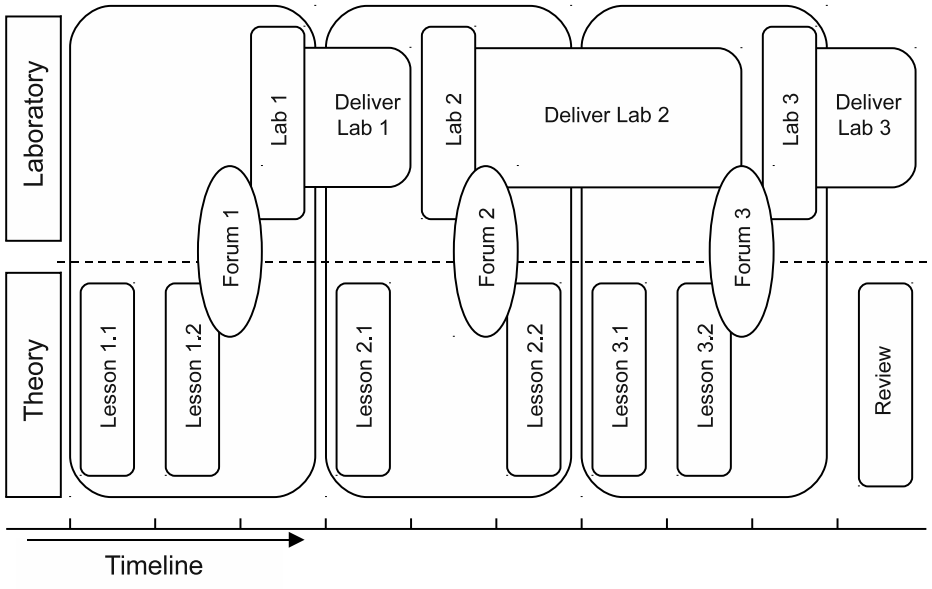


Figure 6.1: Layout of the captured course.

The first step when designing a course in IMS LD is role definition. The specification allows for a powerful hierarchical role definition to group students depending on their tasks. However, the current environment in which the course is deployed required only two clearly defined roles: students and teachers.

In order to capture the course sequence of events, it needs to be cast into the theatre metaphor [203]. In theory, a course needs to be divided into several acts where multiple activities are assigned to roles. Acts allow course designers to deliver activities in a given order, also serving as synchronization points: an act is not finished until all the included activities are completed. In a first approach, it seemed natural to map each theory topic into an act. However, this strategy did not allow for theory sessions of one topic overlapping in time with lab session and student work in the previous topic.

As an alternative, IMS LD provides the so called "activity structures". These structures allow a great deal of parallelism among activities assigned to different users in different roles. Due to the simplicity of distributing persons participating in this course into roles, this approach fit perfectly the course requirements. Also, structures allow hierarchical structures to be delivered depending on boolean conditions at a finer level of granularity. This choice allowed capturing both the sequencing and overlapping requirements for the course.

Activity completion is the most difficult part derived from the underlying asynchronous interaction in the course methodology. As illustrated in Figure 6.2, completing the delivery of a learning object is not effective until the next lab session starts. Managing activity completion was identified then as the key feature in order to fully capture the course learning flow. Fortunately, the Level B functionality in the specification (properties) addresses this issue. The teacher owns a control activity in which completion is managed by a single properties assignment. This is possible due to the support for *imsldcontent* in GRAIL.

The *blended learning* scheme of the course means that not all the activities are done using computer resources. In fact, supporting material for lectures is given in print. Still, these support resources can be specified as environments for the activities included in the UoL.

Each lecture is accompanied by a subset of a *hypertext based book* containing all the theory concepts for each topic. Aside from this resource, environments in the UoL also include the forums required for the asynchronous activities.

Using the notification capabilities included in the Level C of the specification, students can be notified through email every time the tutor activates a new activity in the course. This scheme compensates for the typical disadvantage of a blended learning approach, the difficulty of keeping the students involved in the activities in the course. Upon receiving a notification, students may connect to the platform to find a new activity.

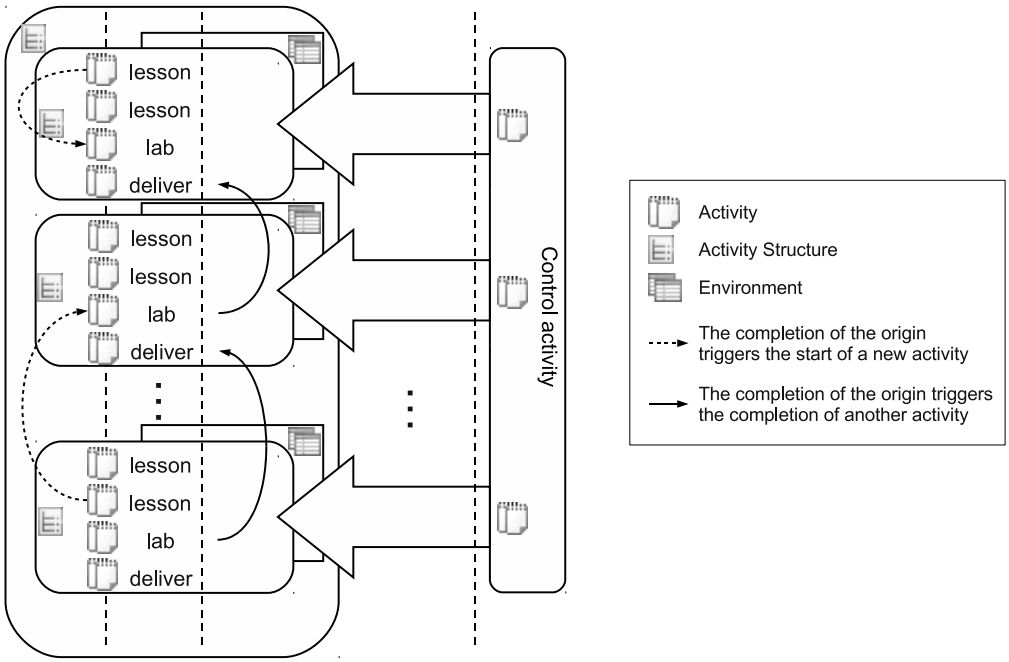


Figure 6.2: Capture of the subject layout in IMS LD.

Deployment issues

Next step in the course replication process is the UoL deployment into the runtime environment. Unless the user interface is well designed, some noise can be introduced in the learning process. That is, if users focus their attention on how to use the tool, they will not learn about the course itself. Obviously, the aim is to avoid this situation.

First issue to take into account is the role assignment. This task must be performed by the course administrator, who registers users in the platform and associate them to the course instance. In non integrated systems, this simple task must be complemented with users' registration into services, implying tasks duplication and requiring coordination in introduced user data. Since the GRAIL application is fully integrated in the .LRN LMS, users' registration is supposed to be already done. Community users are then automatically assigned to the course instance and roles, avoiding coordination requirements. The needed tasks are then reduced to role assignment and a GUI is provided to the task.

Since the application is server side oriented, no special features are required from the client side. A user is able to follow the course with a single browser application. Services,

imsldcontent and all resources are delivered in the same window frame guaranteeing the attention of the students to be focused on the learning objects.

As explained previously, the captured learning flow is taken from an existing subject that is being imparted at the University Carlos III of Madrid. Students are then supposed to already have the know-how of the LMS, requiring no additional effort to start with the course and reducing the noise in the learning flow.

6.2.4 Lessons learned

As discussed in previous sections, IMS LD supposedly covers a wide range of existing methodologies. But this is mostly valid when the course is being designed initially with the specification in mind. In such case, the learning flow is created with the intrinsic limitations of the IMS LD specification. The work discussed in this document was oriented toward capturing an already existing and its learning methodology.

The original course used an in-class learning approach, while IMS LD was designed to be used in purely *e-learning* environments. To reduce the gap between classroom and distance approaches, the captured course was then stated in terms of a more realistic blended learning approach. This change only required a browser as additional tool to be used by students (which they were doing so anyway). The activity tree is partially appearing as the course evolves and students are notified whenever a change is produced. Additional learning tools (or IMS LD services) were included as part of the UoL and were delivered fully integrated with the rest of resources. This decision provided an unexpectedly powerful mean for the teacher to keep track of interaction within the course. It can be concluded that re-factoring the course in this blended-learning paradigm was an improvement for the course structure.

Apart from the required changes due to the in-class nature of the course, all the requisites found in the course learning flow were successfully captured by the produced UoL. Although this fact serves as an example of the expressive power of the specification, the deep understanding required for the translation points in the direction of the features that supporting tools need to provide.

Although the Level B properties were supposed to be used to conditionally sequence the activities in a course depending on student results or choices, in this example they were used for completely different purpose. They were defined for keeping track of the status of all the activities. Teachers then control the learning flow by setting them to the appropriate values. Although correct from the point of view of the semantics, this property usage was a concrete interpretation considered for this example.

Related to this interpretation, an additional difficulty found was the lack of an appropriate authoring tool. The editor used for the capturing process, Reload [86], covers all three levels of the specification. From the point of view of the authoring process, the conclusion then is that the existing course structure was correctly captured, but it required an extensive knowledge of the specification. Such knowledge is far from being realistic in conventional teaching staff members.

On the positive side, obtaining a packaged UoL that contains the complete description of the course structure seems as a promising use of the specification. The overall process had these two advantages:

- Since administrative tasks are done automatically, more resources can be used during the runtime. This includes mail notifications, automatic change of content, etc. These features were not included in previous version of the course due to its high cost in terms of maintenance.

- More effective use of teaching staff time. Since IMS LD captures most of the structure of the course, teaching staff may devote their time to purely pedagogic tasks.

With respect to the deployment phase of the course, the fact that the GRAIL runtime environment is fully integrated with the .LRN LMS allowed for a more compact administration as well as shorter deployment time. User distribution as well as permissions to access the different resources in the UoL were managed seamlessly by the environment. Also, the set of resources available to students was increased by those available in the platform (for example personal file-storage space for each student).

One of the disadvantages of the runtime environment was the difficult task of integrating an agile content modification loop. Creating a complex UoL usually requires producing multiple versions in the authoring space that needs to be packaged and uploaded every time into the LMS and re-deployed. One important feature that was not present at the time of this experience is the possibility to perform changes in the UoL without the need to re-instantiate it entirely.

From the overall experience, the main findings could be summarized as follows:

- The visualization of the activity tree requires the use of a computer with a browser. This requirement is easily achieved but, as a drawback, is not suitable for a lectures-only course. Thus, the course had to be first conceived in a richer blended learning setting. The organization of topics and lab sessions needed to be clarified in order to be specified in the new formalism.

- During the authoring process, the main difficulty encountered had to do with the presence of a highly parallel set of events in which theory sessions were held at the same time as the students working the labs. This session topology required not using the typical play/act contained in the specification and resort to a purely structure based approach.

- Activity termination was another challenge to capture the entire course. By providing special activities, tutors are given complete control as to when a new activity starts and finishes. The possibility of notifying students whenever a condition is satisfied was used to send an email message every time a new activity is available in the course.

- The provided infrastructure for deployment included additional services that enriched the overall experience of the course and significantly reduced the administrative tasks derived from managing a large number of students. The main limitation of the runtime environment was due to the difficulty of modifying the UoL while is being processed.

6.3 Collaboration on distance scenarios

The successful results and the experience gained from the process described in Section 6.2 suggested the development of new scenarios modelled with IMS LD, this time including the enactment phase. Thus, the next step in the study of the specification was the creation of a learning flow where the pedagogical method highly relies on collaboration among students. Two experiences were developed with the goal of understanding the capabilities and limits of the specification in distance situations. The fist of these experiences was a pilot enacted by volunteers, while the second one was populated by students of different degrees in higher education institutions. The steps required in such experiences are: the course design where the collaborative pattern is decided; choosing and deploying tools to support the model; its translation into a IMS LD representation; and finally, the assignment of students to the different groups and related tasks. This dissertation mainly focuses on the second and third steps.

This section is devoted to describe the goals, activity sequence, deployment details and lessons learned from the two experiences. They were implemented in the context of the

Mosaic project [204] and involved participants from three different Spanish universities: University Carlos III of Madrid, University of Valladolid and Open University of Catalonia. The designed workflow was a combination of well-known collaborative patterns such as jigsaw, peer-review, think-pair-share and pyramid [205]. These patterns have been extensively used and analysed in face-to-face scenarios, but impose several difficulties on distance learning. The deployment and enactment of the described experiences allowed identifying the needs of such collaborative models on distance scenarios, at the time that provided an insight of how IMS LD can be used to succeed in the enactment process.

6.3.1 Description and objectives

Despite the two described experiences differed in several aspects (e.g. the participants' profile, the context of the experience or some details in the learning flow), the core elements of the learning flow were the same in both cases. The main difference is that the first experience was a pilot program developed as a first approach to the problem, while the second one was an authentic experience where the identified flaws of the pilot programs had been improved.

Goals

The deployment of the experience here described required multiple resources, coordination between peers and the use of different technologies. One of the aims of the experience was to test the functionality of these technologies in a real scenario. It follows a description of the objectives of the experience.

A wide range of techniques for collaborative learning, tips for improving the learning experience with computers, and tools are contained in the CSCL research area. However, there is a lack of documentation of pilot programs where collaborative schemes have been put in practice on distant scenarios.

The presented experience contains a high level of interaction because it was derived from an in-class scenario. Deploying this learning flow on distant scenarios shows how the interaction between peers is possible and how the coldness of the environment has influence on learning. The aims were to identify where the most relevant difficulties are, how the students react to the experience, and what are the suitable technologies to perform the deployment successfully.

IMS LD is expected to cover a wide range of pedagogical models, including collaborative schemes. It has been shown that collaborative flow patterns have several characteristics that made of their development a difficult task:

- A collaborative flow usually implies the participation of different roles. The language must provide a method to differentiate students between them. Capability of grouping must be also provided.

- Interaction between peers is highly required. A complete course description must specify how these interactions occur.

- The learning flow is susceptible to change, depending on the availability of students and many other factors. The flow design may take into account these adaptability issues.

The work presented in [206] discusses the theoretical completeness to capture collaborative patterns. The experience described in this document shows the IMS LD capabilities from a practical point of view.

Finally, an experience in an authentic scenario allows evaluating GRAIL from the usability point of view, which is one of the factors that may affect the success of such type of experiences.

The collaborative learning flow

Depending on the topics studied by the participants on each case, there were some adjustments on the learning flow but they both shared a common structure, which is described as a combination of collaborative patterns. According to the needs of the learning flow, the teacher was included as a tutor, a facilitator of learning, but not as a lecturer. The common structure of the learning flows can be described as follows:

1. An open-ended problem is stated. Also, each student is identified with a different shape and colour (i.e. red-triangle, red-rectangle, blue-circle, etc.)

2. The students are grouped in the so called *expert* groups (students with the same shape), which means that members of the same group face the stated problem from the same perspective.

3. A problem description and the corresponding reading resources are delivered to the students, where all members of the same group receive the same resources. This phase is individually performed and the students produce a document with his/her proposed solution and the extracted conclusions.

4. The expert groups meet and they peer-review the documents or teammates. Then, they discuss for the optimal solution and write an agreed conclusion extracted from the discussion.

5. Expert groups are dissolved and the students are grouped in *jigsaw* groups. That is, all team members are identified by the same colour. In jigsaw groups, each member has faced the problem from a different perspective so there is positive interdependency among peers.

6. Within jigsaw groups, the students present their solutions to the problem and discuss for a new solution that considers all the different approaches. They produce a document that states their agreed conclusion.

The details that correspond to the pilot program and the authentic experience are given in Subsections 6.3.2 and 6.3.3, respectively.

6.3.2 Proof of concept: enactment with experienced users

This first experience, consisting on a pilot program that tested the feasibility of the scenarios, was developed in the context of a Master Thesis, part of the doctorate program in the Telematics Engineering Department of the University Carlos III of Madrid [207] and the above-mentioned Mosaic project [204].

Details of the scenario

Three high level educational institutions participated in both course design and deployment. Four students at each institution followed the course. These students were selected according to the following criteria:

- The course was not part of a degree. The students were volunteers and no reward was offered. This student profile was selected because the experience was conceived as an experiment, so this condition minimizes the possible administrative conflicts in case of severe deficiencies.
- High level computer skills were required. The experience tested if the technology is ready to be used on the scenario focusing on having a proof of concept, without evaluating usability problems.

In this context, the course topic was *Grid Computing*, and it was offered to twelve PhD students in computer science.

Adapting the flow to the scenario

Each of the twelve students is represented as a pair shape-colour, which corresponds to the expert and jigsaw group, respectively. As depicted in Figure 6.3, three shapes and four colours were defined. That is, there were three expert groups and four jigsaw groups. The activity sequence is then described as follows:

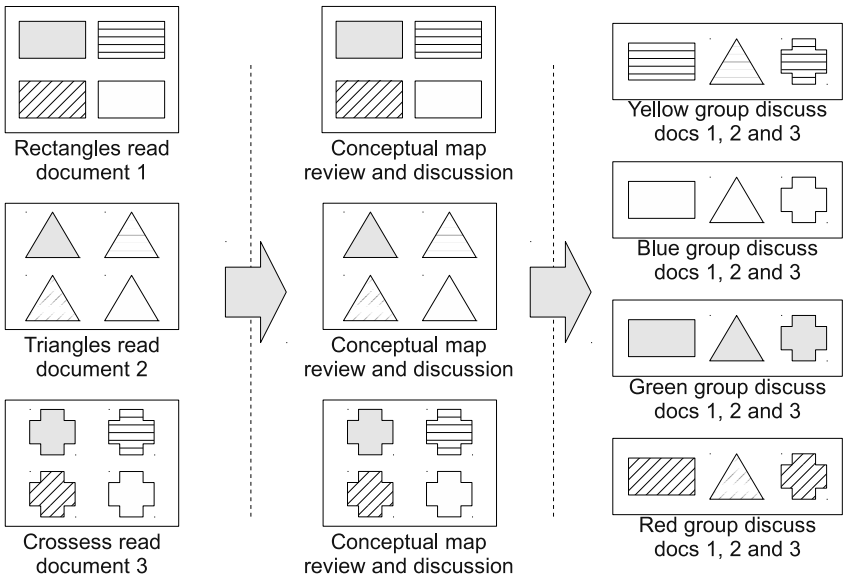


Figure 6.3: Collaborative learning flow enacted in the pilot program.

1. A different article (a research paper about grid computing) is delivered to each *expert* shape. Students have to read it and create a conceptual map with the topics discussed in the document.
2. All conceptual maps are reviewed by expert peers, culminating in a group discussion which improves the paper understanding.
3. Playing the role *jigsaw*, conceptual maps review takes place. In this case, each student reviews jigsaw peers conceptual map, corresponding to the topics discussed in a document different from the document read by the reviewer.

4. The discussion is repeated within a bigger group, where more jigsaw teams are involved. This step is repeated several times increasing the number of participants until all the students are discussing together.

In a distant scenario as the one presented in this experience, awareness needs to be specially addressed. The students must be clearly told where (understood in the sense of "in which activity") they are and what they have to do at this point. Since different students are assigned to different tasks and must collaborate with different peers, they must be provided with different assignments. That is, the course flow has to show the same learning path to all students, despite not showing the same content at each activity. This is not difficult on in-class collaboration, but turns into a problem on distant scenarios.

The used model requires high level of interaction between peers. That is, they have to exchange documents, communicate opinions, scribble on a shared picture, etc. Each of these interactions requires the support given by a different tool that cannot be managed with IMS LD. These tools that are in the learning model but whose location and configuration cannot be determined in the IMS LD course description, are henceforth called external tools or services. For the purpose of this experience, services were X-based[1] utilities that were accessible through a VNC server.

Deployment and enactment

The Unit of Learning was deployed and enacted in GRAIL. Such UoL was divided into 6 different acts, corresponding to each session of the course. The main IMS LD elements used in the UoL creation description are:

Roles. During the course, users play different roles depending on the running activity. As discussed before, they have to be experts on a topic, members of a jigsaw or members of a group discussion. To group students in such a way, sub roles of the main (and mandatory) student role were created representing all the different behaviours in the course. On the other hand, teachers are also participants of the course, having their correspondent role and related activities.

Properties. Roles are not enough when grouping people, they are not appropriate when different groups has to be built in the same role. Groups can be created, but no special operation can be done with them.
The aim is to provide each one with a different assignment to each group. Local personal properties, used in combination with conditions, were defined to manage this function. Thus, shape and colour of each student were tested before assignments delivery, in order to accommodate content to users. The activity tree is designed for all participants but each of them only receives his/her part. Each role will receive a different file in a given activity. Inside files, content may be common or may change for different shapes and colours.

Environments. When a service is used in a UoL, it has to be defined inside an environment, which will be related to one or more activities. Following this working scheme, external services were linked from the environments. Learning objects consisting in a simple HTML file containing the URLs required to access the collaborative tools. Note that this is a one-way loosely coupled system where no runtime information exchange is possible.

[1] Linux tools that offer a graphical user interface.

The collaborative tools were hosted externally. It is not easy to find a tool that meets all the requirements for a given activity. However, any software can be used as a collaborative one by sharing it with VNC software. Since students are working in groups, each group needs its own instance of all required tools. Moreover, groups change during the course, so tool instances must be available for all possible scenarios. As a result, each used tool must be instantiated and shared a large number of times.

But this approach produces an administrative problem in the database. In order to provide access and rights for different users, both IMS LD and VNC servers are requested to store and manage groups' information. The system therefore is prone to misconfiguration problems. This architecture is an ad-hoc solution for the given case, but it is not integrated in the UoL, reducing its reusability. The two implied systems are one way linked, as shown in Figure 6.4.

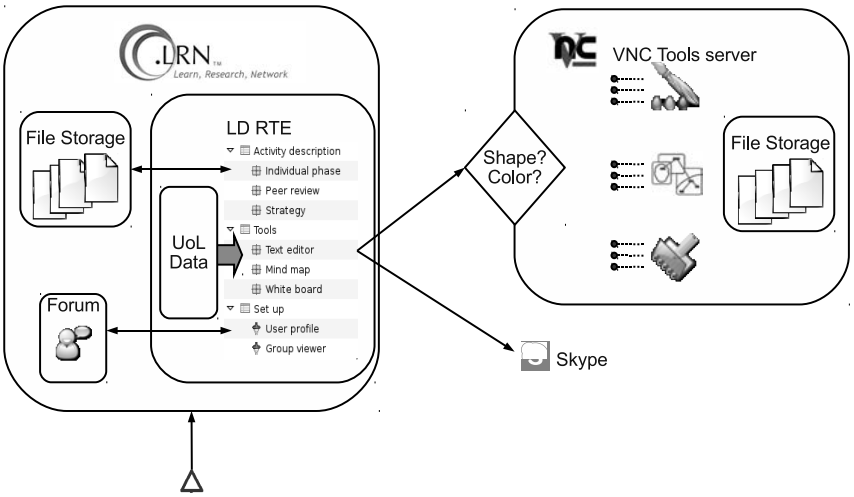


Figure 6.4: Link between VNC services and the IMS LD player for the pilot program.

Identified problems

The learning experience described on this document has provided quite relevant data to qualitatively analyse how IMS LD can be used on the collaborative field. The experience was considered a successful proof of concept. The course was designed, deployed and enacted covering the initial requirements. Thus, it has been shown that is possible to package a complex collaborative learning model in a UoL within the IMS LD specification. It also has been shown that existing tools provides a framework for distant collaboration. The course success must be understood on the following context: all students have high level skills on computer usage so usability lacks had less impact on the normal course flow, if compared with regular students. This scenario is therefore not suitable for testing usability issues on the used platform.

The authors of the UoL were expected to manage a large amount of details, making the designing process a difficult task. Furthermore, the available authoring tools[2] required a deep understanding of the specification. In the described experience, some mistakes appeared in

[2]Reload was used

the course during its enactment (being visible to students) due to the cognitive overload imposed by the authoring task.

During the authoring, roles were used to model the grouping requirements of the collaborative flow. The students needed to select, at the beginning of each activity, the role they were going to play. This fact, which is the intrinsic behaviour when a student belongs to different roles, caused confusion in the students that sometimes were not aware of their corresponding activity. The authors concluded that the use of properties to model groups is more flexible for authors and less confusing for students.

The given course uses external services as a way to provide support for collaborative tools. Regarding this fact, some problems appeared:

- The use of two servers derived in the duplication of administrative and maintenance tasks. The need to specify how servers may interact made the deployment slower because it had to be done manually. As a result, the reusability of the experience was severely affected.

- It was not possible to package the external services within the UoL. Thus, the services were accessed through links manually placed during the authoring phase. The straightforward implication was that authors needed to know the details of the enactment phase, which does not fit the course life-cycle of IMS LD. Another implication is that these URLs were not reusable, and therefore the course cannot be used more than once.

- Due to the scenario requirements, the students' data were stored both in the VNC server and the IMS LD one. This fact required the synchronization between servers and required a significant effort to duplicate resources. Noise in the learning process was also introduced, since the student managed different resources in different servers. A method for automatic data synchronization would solve these limitations.

The documents produced as the activity output were used as the input of the following ones. However, these documents were elaborated at the external services so they could not be considered within the IMS LD flow. As a result, the learning flow was unable to orchestrate the artefacts' flow, which was manually performed. The analysis of the artefacts' flow in this case of use was the matter of research of the group of the University of Valladolid that participated in the experience [116].

Finally, a method to manage unexpected events was required during the enactment. It is an important drawback since course designer and teacher are roles played by different people. It is usual to find that the teacher requirements do not match exactly with the designer ones, and in this case the teacher has no capability to react. Furthermore, given that the success of collaborative learning flows relies on the participation of team members, the absence of one of them may result on the incompleteness of the activities. IMS LD lacks of means to manage these unexpected events and the consequence is the poor flexibility that the course had.

6.3.3 The experience in a regular course

Also in the context of the Mosaic project, and within the participation of the three educational institutions that participated in the previous experience, the learning flow depicted in Section 6.3.2 was enacted in an authentic experience and it was included as part of their regular undergraduate programs. The course participants were students from different degrees in the involved institutions. The experience described in this section was published in [115, 208].

Details of the scenario

A total of 31 participants took part in the experience: 27 students and 4 tutors. Participants at University Carlos III of Madrid and University of Valladolid were undergraduate students of the Telecommunications Engineering degree and for them the course was optional. On the contrary, for the students of the Open University of Catalonia, undergraduate Computer Science students, it was part of a larger non-optional programming course.

Students were divided into three groups, each of which participated in a replica of the course. The learning structure was then defined for a group of nine students, and three identical instances were enacted independently from each other. In order to maximize availability, tutors were not assigned to any particular instance. Henceforth, the described learning flow refers to a group of nine students.

Since all participants were computer engineering students, they were assumed to have basic computer skills. It was also assumed that students from the same institution had similar programming skills, and therefore three different profiles were created. The motivation for this division is that collaboration with different skilled peers from a different institution may increase the discussion effectiveness [209]. It is worth mentioning that one of the institutions participating in the course, the Open University of Catalonia, follows a distant education paradigm for their regular courses. As a consequence, students from this institution have different scheduling requirements, and in most cases their involvement in the activities is purely asynchronous. This fact had an impact on the learning flow, due to the impossibility for groups to synchronously develop the discussion tasks.

On a collaborative learning model, students are supposed to be active participants. Success highly depends on student motivation [210] so the course topic must be carefully selected in order to ensure positive participation. The described case was based on the study of Drupal, an open-source Content Management System written in the PHP scripting language. This topic was selected based on the following observations:

- PHP is one of the most popular scripting languages used for fast Web development, and yet, it was not fully covered by any of the regular undergraduate courses. Therefore, a course on the subject was expected to be attractive for those students interested in improving their curricula.
- The Drupal platform is a real-life application that allows students to modify current features and see the effects quickly in a mature environment, thus opening the possibility of new developments.
- The learning model for this type of platform requires certain degree of self-study. Students are supposed to search for additional documentation aside from the material provided by the tutors. Drupal has a large user community where information is complete and well-structured.

Adapting the flow to the scenario

The proposed learning flow for the course is based on a combination of collaborative learning flow patterns, or CLFPs [211] . The overall structure leads user activities to a global goal, being problem-based learning [212] the underlying strategy. An open-ended problem is presented to students while tutors are facilitators of learning. The goal is not the solution itself, but the acquisition of problem analysis skills on the given environment.

In the described case study, the activity sequence is given as follows:

1. There is an initial individual study stage, where students examine provided documentation. The problem is not presented yet. The requested output -a brief summary of the readings- creates a link with the next activity.

2. In a second phase, students work in three-member groups to peer-review their summaries and agree on a common conceptual map of the discussed ideas according to the "Peer-reviewing" CLFP.

3. Once students receive the statement of the problem, a group decision sets the development strategy, which will be put in practice individually. All groups work on the same problem. However, each group assignment is focused on a different aspect of the solution. Students become then experts of a given aspect.

4. Next, students are re-organized into jigsaw groups, where members are experts on different aspects, promoting positive interdependency. Individual coding solutions are now reviewed by the group.

5. In the last activity, according to the "Jigsaw" CLFP, developers have to work in their jigsaw groups and join their partial work to obtain a complete solution.

An overview of the learning flow is depicted in Figure 6.5 for one particular participant. The "shape" of the student (e.g. triangle) represents his/her expertise while the "colour" (e.g. white) indicates the jigsaw group he/she belongs to.

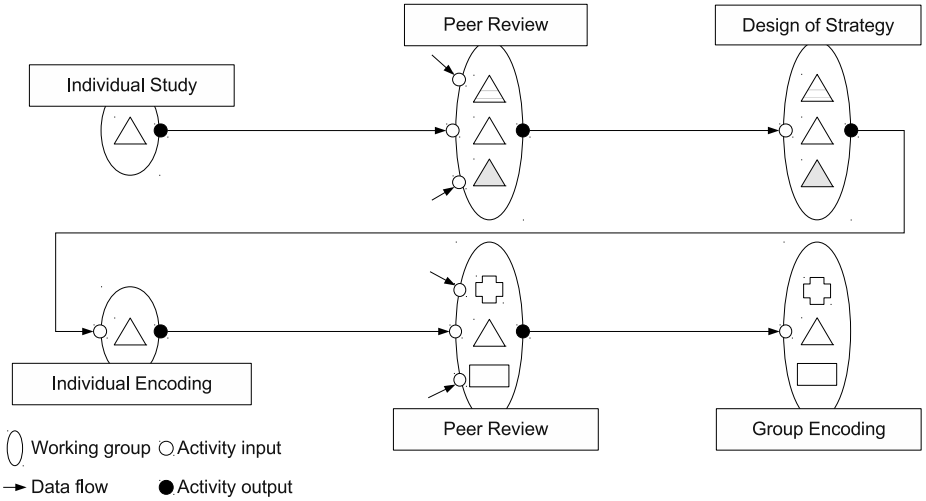


Figure 6.5: Example of learning flow for the student "red-triangle".

In all the activities, students are encouraged not to use any extra time to finish them. The reason of this restriction is to avoid excessive distance between user skills and provide a framework where peer-review is effective.

Deployment and enactment

Due to the high interaction level required in the activities, collaborative learning models are typically deployed in a face-to-face environment. When applying this model to a distant scenario, as done in the described experience, both student expressiveness and model flexibility are reduced. The course deployment phase had to be adapted to the scenario in order to minimize the above problems. The use of supporting tools allowed the inclusion of certain degree of flexibility in the model without conflicting with intrinsic constraints [213].

Tools for collaborative work allow multiple users to synchronously manipulate different resources, providing a framework where collaboration is more fluid. However, student profiles did not guarantee availability for synchronous sessions. As a consequence, an average of one student of each group was supposed not to use the synchronous collaborative tools.

To support different profiles, the use of both synchronous and asynchronous tools had to be merged in the activities: discussion recordings were used as input for asynchronous students, who were assigned to review discussed topics and arguments. A forum is used for the review, so that a parallel discussion was encouraged. The tools used in the experience are depicted in Figure 6.6 and summarized in Table 6.1.

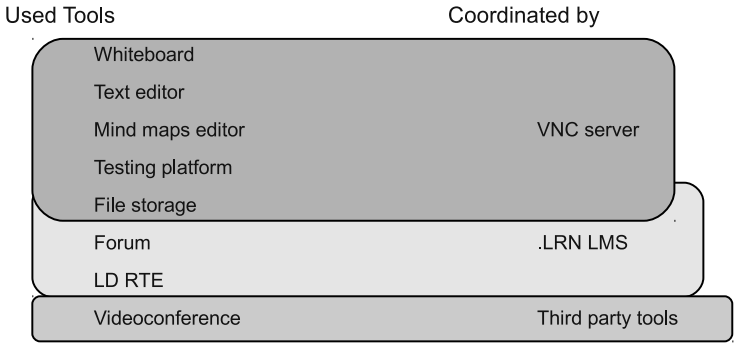


Figure 6.6: Tools that supported the experience and their corresponding platform

The joint use of synchronous collaborative tools and asynchronous communication facilities was intended to reduce the gap created by the distant scenario and allow students with different profiles to work together. This approach had an impact on the learning model, so that the authoring of the UoL had to consider this fact. The most relevant parts of the created UoL were:

Group set-up. The high level of interaction required in the model is reflected in a complex grouping model. Groups must be created and dissolved during the course, and the supporting system must be flexible enough for this management. Taking into account the lessons learned from the previous experience, working groups are related to LD property values. Thus, group behaviour is defined at course instantiation and the tutor can modify it during the course if required.

Adaptation issues. Synchronous collaboration-based activities are not always suitable for all students, especially when they are from different institutions. In our case, one of the participant institutions follows a distant learning model for regular courses so in most cases students schedule does not match with peers. It is therefore not possible to meet all group members at the same time. The asynchronous students are given an adapted assignment, where tasks are modified to match the case. The students are also capable to modify their own profile, adding more flexibility to the model.

Despite the described flexibility in the model, supporting software does not allowed changes in the learning flow (run time script edition) once the course has been deployed. This was especially relevant in a scenario where the design task was considered error prone [116].

The system architecture that supports the model is composed of two different servers (Figure 6.7). The LD server contains the learning flow and delivers the assignments to students. These assignments are linked to VNC based collaborative tools, hosted on a second

Tool name	Description	Supported activity
Kolourpaint	KDE default image editor. Allow scribbling as in a whiteboard	Collaborative whiteboard to support the arguments in group discussions.
Kate	KDE plain text editor	Collaborative creation of documents, used to write the minutes of the group discussions.
CmapTools	Mind maps editor	Used to elaborate the output of some of the activities.
Drupal	PHP based Content Management System.	Used by the students to test the correctness of their develops. Students were able to share (or not) the system the produced.
File Storage	Web based repository	The output of the activities was stored there, so it acted as a *de facto* data flow manager.
Forum	Web based forum	Used to asynchronously ask for support in the learning activities or with the use of the platform
GRAIL	IMS Learning Design player	The students accessed this tool to retrieve their corresponding activity description.
Flashmeeting	Web based videoconference system	Supported the synchronous work sessions.

Table 6.1: Summary of tools used during the experience.

server. Relevant data are automatically synchronized between servers to avoid database inconsistencies.

Analysis of the experience

Since all requirements in the course design and deployment were covered, the experience was considered successful. The feedback obtained from the participants offered multiple observations that allow a detailed analysis of the enacted course.

The data gathered from the study, summarized in Table 6.2, was collected through questionnaires that participants filled at the beginning of the course (to know their expectations and background), as well as after the experience was over (to obtain their viewpoints). Some additional evaluation data such as messages posted in the course forums and systems logs were also available. Questionnaires had a mixture of quantitative and qualitative questions and their analysis followed the principles of a mixed evaluation method [11].

One of the main conclusions is that supporting technology was not mature enough. Tools supporting communication in collaborative environments are mainly used in pilot programs. Very few experiences where a complex model is deployed in a real-life course are found in the literature, and therefore, there is no best practice guide to face the difficulties that

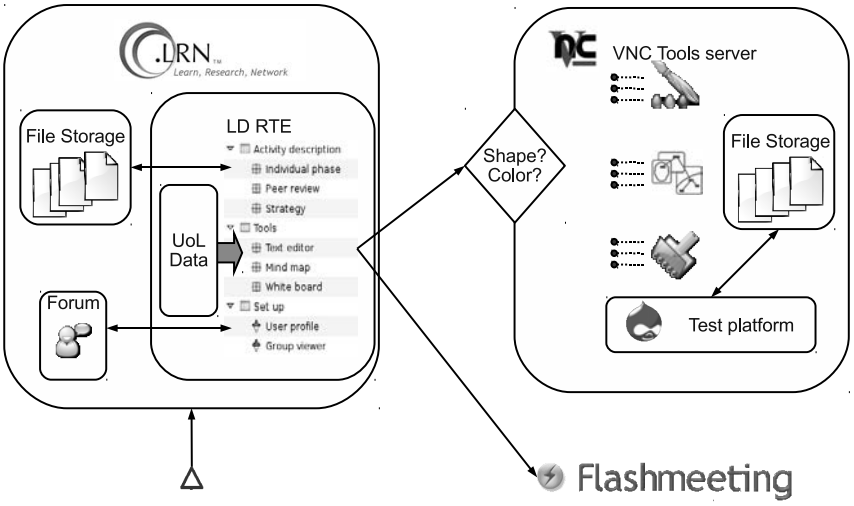


Figure 6.7: Architecture of the system that supported the experience.

Data source	Description
Quest	Questionnaire answered by the students after the course
Forum	The forum of the course. The students were encouraged to post questions regarding the course and/or the method

Table 6.2: Data sources that allowed the experience analysis.

appeared at runtime. Participants ranked an average of 2.20 (in a Likert-scale of 0 to 5) the technological support in the course. This result is mainly due to the lack of robustness of the tool prototypes. Nevertheless, several participants recognized the added value and potential of the presented technical solution for supporting distance collaboration. For instance, one of the participants said: *"I think it is a very useful way of working with participants who are not located in the same place, in spite of the failures in the technical support"*.

Although the tools covered all the required functionality to support the course, they failed to address some basic usability issues. The result was an environment that is fully functional but not user-friendly enough for a real case. In practice, it took some time for students to get used to the interfaces. This had a special impact on the experience because of its time restrictions. The solution for this problem is to schedule a training session with the students before the course starts. However, the most effective solution would be to use tools with intuitive interfaces providing a low adoption threshold. In this sense, even though it was conceived as part of the architecture, participants did not perceive the various distributed applications as an integrated solution (an average 1.58 in a 0-5 scale). Additionally, some opinions pointed out this low level of integration as a source of difficulties. For example, one the participants stated that *"... you did not have the freedom to go from one application to another and, furthermore, you perceived them as something external..."*.

Having students from three different institutions increased the existing gap between

profiles. Students' initial skills in the course topic were significantly different. Some participants argued that the course level was too high, while others performed the activities easily. Overall course difficulty was ranked with an average of 3.40 (in a 0-5 scale). Far from being transparent to students, this fact had a negative impact on their motivation.

A summary of the findings, the data that supports this finding, and the source of these data are presented in Table 6.3.

Finding	**Supporting data**	**Source(s)**
Supporting Technology is not mature enough	- "Technical failures derived on delays when doing activities"	[quest]
Students did not get a perception of having an integrated solution	- "You did not have freedom to move between tools. Their slowness make them appear as something external"	[quest]
Collaboration was perceived as a positive factor on a course	- Opinion about collaboration level with peers [0-5]: $3,49$	[quest]
Asynchronous support widely used. Users missed synchronous support	- 195 participations on forums - "Sometimes we had doubts that need to be solved just at the moment"	[forum], [quest]
Complexity of learning flow not appropriate for students experience	- "Did peers with the same shape receive a different statement? Or... is there any mistake?" - "I think groups must not change during the course"	[forum], [quest]

Table 6.3: Main findings extracted from the analysis of the experience.

6.3.4 Lessons learned

Replicability

The course flows described in Sections 6.3.2 and 6.3.3 were deployed in different academic course, with a time lapse of one year between them. In such scenario, where both course flows share a high amount of elements, the reusability of the model plays a relevant role in order to provide sustainability to the overall model.

The total cost of the first edition is devoted to authoring (design of the pedagogical model and development of the material), deployment (configuration of the supporting platform) and enactment (tutoring and supervising the course). In the second edition of the course, the cost of authoring is reduced to the development of the material. However, there was almost no reduction in the time used for deployment and enactment.

The researchers that participated in the two course editions agreed on saying that the deployment of the learning experience was time consuming and required a high effort to be implemented. They also said that there was no significant reduction of the required time on the deployment of the second edition. Thus, there was a high cost on the course flow replicability.

The main cause of this high cost is the use of external services:

- The configuration of all the services is performed manually. The requirements of the scenario imposed the need to configure as many service instances as users are in the course.
- The course material contains the URL of the external services, so that it must be modified each time the course is replicated.

The lack of replicability was identified as one of the major limitations of the proposed model. As a consequence, one of the goals of the subsequent experiences was to increase the replicability of the proposed model. This is the main matter of research in the experience presented in Chapter 7.4.

Flexibility

In collaborative learning situations like the ones presented in this chapter, the pedagogical success depends on how implicated are the students within the activities. Strategies such as the promotion of a positive interdependence or the use of innovative technologies try to increase students' motivation and therefore increase the likelihood of success.

However, even with motivated students, the absence of one group members is very common when the number of participants is high (27 is considered a high number for the model). Such absence is, by definition, an unexpected event. Other sources of such unexpected events are mistakes in the course material or students who do not store their output where they are supposed to do it.

The capability to reorganize the course so that an activity still makes sense without all teammates, to correct mistakes in the course or, in summary, to handle unexpected events, is called flexibility. IMS LD provides a certain level of flexibility, which could be called *pre-programmed* flexibility. That is, rules can be created during authoring so that certain situations result in the adaptation of the activity. However, this is not a real flexibility because it is only useful with situations that can be envisioned by course authors and they are not, by definition, unexpected. In scenarios that follow the course flow imposed by IMS LD, it becomes apparent the need of more flexibility during runtime.

The flow of artefacts

The IMS LD specification is devoted to orchestrate learning flows, but it is not able to define mechanisms to orchestrate data flow.

On the one hand, output files could be attached to IMS LD properties so that they allow exchanging the documents among peers. However, properties lead to low-flexible and error-prone models whose administration is difficult and time consuming. On the other hand, IMS LD cannot manage artefacts generated in external services. That is, if the data flow is outsourced, its relationship with the activity flow is broken and there cannot be coordination (at a reasonable cost) between the two flows.

This problem is still an open issue and, despite it is highly related with it, is out of the scope of this dissertation. More details on the study of the problem and possible solutions can be found in [116].

Tool integration

The distant nature of the course reduces student expressiveness during collaboration and this problem has to be diminished by the use of supporting tools such as a shared whiteboard where drawing can enforce student arguments given by videoconference. However, such tools are not supported by IMS LD. Courses can make use of external tools to support the learning activities, but these tools cannot be integrated in the course package.

As a result, the external services and the course flow need to be configured separately. This problem affects the abovementioned replicability of the courses. Furthermore, the high cost of the scenario configuration makes difficult to pay attention to other aspects of the experience such as the perceived integration of the overall system and its usability.

Software usability

The final results of a computer supported learning experience depend on the functionality of the software and its usability. The former refers to the actions that the course participants are allowed to perform with the supporting platform, while the latter is related with the difficulty that the user will find on performing such actions.

Usability was one of the weaknesses of the presented experiences. This fact has negatively influenced the student's perception of the platform and also their learning on the studied subject. Further experiences should pay more attention on usability or, if it is not possible, provide a training session prior to the execution of the experience.

6.4 Integration with *Mupple*

The experiences described in Sections 6.3.2 and 6.3.3 clearly show the need of case-specific tools to support the activities performed in a distance delivered learning flow. The Web 2.0 provides a vast catalogue of available supporting tools where is not easy to select the most appropriate one for each case. Furthermore, such tools are integrated on students' everyday life so that they have their own personal preferences, which usually differ from teacher's preferences. The unfolding landscape is that of a community of users with a highly personalized environment prepared to interact with a large number of available services. The success of personalization platforms such as iGoogle, Netvibes, Pageflakes (just to mention a few) together with their potential to increase the effectiveness of learning experiences sustains this claim. Mash-ups, Web applications that combine data from different sources, are steadily gaining acceptance on the *e-learning* context.

But with the possibility of combining a set of services comes the challenge of inserting such applications in current specifications used to formally capture the interaction and resources needed in a learning experience. The IMS Learning Design specification [6] (henceforth IMS LD of simply LD), provides a language capable of defining the interaction that takes place in a learning environment. While the framework claims to be generic, it is oriented toward capturing interaction at the level of the different activities within a learning experience. Furthermore, although the specification includes a property-based adaptation paradigm, these properties need to be defined at design time and therefore typically refer to aspects that are statically included in the unit of learning.

The emerging scenarios provided by the use of mash-ups are merely environments that do not necessarily provide learning without an orchestration that provides a pedagogical sense to the activities. It therefore appears the need to combine these scenarios with the description at a higher level of a Unit of Learning. The solution here explored is to combine both paradigms: on the one hand, the Learner Interaction Scripting Language [214] (henceforth LISL) is a language defined to create a learning environment as a Web-application mash-up:

an aggregation of user interfaces from different tools that are combined to achieve learning outcomes. These mash-up based environments are then placed in Unit of Learning IMS LD environments, providing tools to perform course activities. It is here described how such integration has been done in Grail [10]. As a result, a Unit of Learning is used to capture the higher level organization and resource of a learning experience, but at the level of an activity, students are offered a highly flexible learning environment described in LISL.

This section includes a discussion of how these technologies can be combined. The discussed approach was implemented so that the supporting tool was available, but the developed example Unit of Learning consisted just in a proof of concept that was not deployed on a real scenario. The discussion is here included because the lessons learned from the experience allowed determining which the requisites of GSI are, so that it played a relevant role in the presented dissertation. This work was presented and published in [215].

6.4.1 Flexible learning scripts

The term "script", when applied in a pedagogical context, refers to the method to structure a learning process [211]. Course participants are guided through the flow of activities previously defined by the authors of the script. The level of coercion in this script will influence the course success. Over-scripting, or a too detailed set of steps that must be followed, may reduce course effectiveness, while a too flexible scheme may not produce the expected interactions [75]. The trade-off then is to create a set of instructions detailed enough to guarantee a successful learning experience, yet leaving room for certain flexibility.

According to the lessons learned in previous experiences (see Section 6.3.4), the attitude of students cannot be predicted during the authoring phase, which may lead to different unexpected situations. To minimize the problem, course participants can be monitored at the enactment phase, especially if the course is being taught in the context of a LMS. In the ideal case, the runtime environment should provide tools that enable the reaction to unexpected situations. This capability of changing the course behaviour while is being enacted is known as *run-time flexibility*. The relevance of such flexibility on scripted collaborative learning is analysed in detail in [74].

The type of modifications that are typically required in scripted learning flows ranges from structural changes, content modifications and dynamic group management. Based on these different requirements, the required flexibility in a learning environment can be classified in different ways.

The first classification criterion is the scope of the applied changes. Flexibility can be said to be at macro or micro script. The macro-script level refers to the high level learning flow: the activities that are present in a course, how artefacts are produced and consumed by these activities, how and when participants can be monitored and evaluated, time scheduling, or any other factor that affects the course as a whole. The second type, micro-script flexibility, refers to activity-centred modifications. For example, how a single activity can be properly performed, including the tools to be used, the available content and expected interactions among peers and resources, etc... Thus, flexibility at the micro-script level allows re-defining how a single activity is performed. A learning script is composed by an overall structure (macro-script) and the particular details of the activities (micro-script). A flexible learning environment should therefore pay attention to these two levels of flexibility.

A second classification criterion is the course participant that will need to perform the modifications, teachers or students. Teaching staff (who are not necessarily course authors) are in charge of supervising that the activities are being doing as expected in the original script. If they do not, teachers might need to perform changes (macro or micro) to guarantee that the objectives can be reached. One simple example is the extension of the deadline for students' submissions. Students are not usually allowed to modify the conditions of

the learning flow, but they might need other type of modifications. For example, some pedagogical approaches encourage students to select their preferred topics to study. Another example is the customization of their learning environment, including the specific tools to use in the activities and the look and feel of their personal environment. The learning environment should offer certain degree of flexibility to customize to certain extent the learning environment without interfering with the applied pedagogical method.

6.4.2 Integration of Learning Design and LISL

In computer supported learning scenarios, scripts can be formalized using a modelling language. All the interactions of a learning flow are then captured in a single file, which can be deployed in a compliant platform, where the enactment takes place. It follows an analysis of how runtime flexibility can be provided combining two different scripts: IMS LD and LISL.

Macro-scripts

The IMS LD specification provides a framework where a wide range of pedagogies can be expressed. It is therefore a specification that matches perfectly with the concept of macro-scripts. Course structural changes during runtime can be managed by a proper use of LD properties, but they must be anticipated during the design phase. Taking advantage of this potential use of properties, IMS LD offers a significant level of runtime flexibility [115].

From the teacher point of view, content can be made visible depending on the value of a given property. The modification of such property can be manually performed (with a *monitor* service) or can be triggered by events such as the completion of a certain activity, the achievement of a given mark or simply the waste of a pre-defined time lapse since the beginning of the course. Property-driven modifications can be therefore easily applied during the enactment. Group management can also be captured by properties, allowing dissolving and regrouping students if it is required by the collaborative settings.

Typically, the described learning flow cannot be modified by students, but they can take decisions that affect their learning path. This is the case of profile-based statements: if students can modify their own profile, they will be able to select the kind of activity they prefer, without breaking the course constraints. This is also possible in IMS LD with the use of properties.

Although certain degree of flexibility can be achieved during runtime, it is at the cost of capturing these changes with numerous properties and conditions. The main consequence is that all possible changes need to be anticipated during the authoring phase. This aspect hinders significantly the possibility of performing fine-grained modifications.

Micro-scripts

The IMS LD specification is clearly oriented to macro-script creation and therefore is not appropriate to scaffold the interaction process that occurs within an activity. A new approach is required that allows participants to select their preferred environment for the required activity. The LISL language can be used to create micro-scripts that create, manage and maintain fully-customized learning environments, and can be used to offer flexibility to students while enacting an activity [214].

The LISL language allows defining **actions** to perform in a given activity, **objects** that are produced, modified or consumed by actions, and **tool** to perform requested task. Objects can be defined by a URL, and tools must be related to a URL. LISL statements are close to natural language, so that they can be read as follows: “perform action A on object B using tool C”. When the script is played, all required tools are opened in the working environment with mash-up visualization techniques.

Simplicity of LISL script creation and modification was one of the design premises of the language. As a result, the personalization of working environments becomes a really agile task that can be performed by course authors and modified by students during runtime.

Implementing the Mash-up Integration on Learning Design

In order to integrate a LISL-based mash-up with a IMS LD player, the used mash-up templates need to be simplified to achieve a plain look. This interface simplification and the use of technologies such as AJAX helped to perform the integration very intuitively and in an unobtrusive way.

On the side of the IMS LD player, the mash-up is referenced as a regular learning object within the environment of a given activity. Thus, the LD player will provide an independent highly customizable learning environment while remaining unaware of the level of freedom provided by the mash-up.

6.4.3 An Example Unit of Learning

In order to provide a proof of concept of the combination of the two paradigms, a simple UoL that contains the previously discussed features was created and deployed in GRAIL [10]. The .LRN LMS also provides a LISL interpreter called *Mupple*. Ad-hoc modifications were performed so that GRAIL was able to include Mupple as one more supported service.

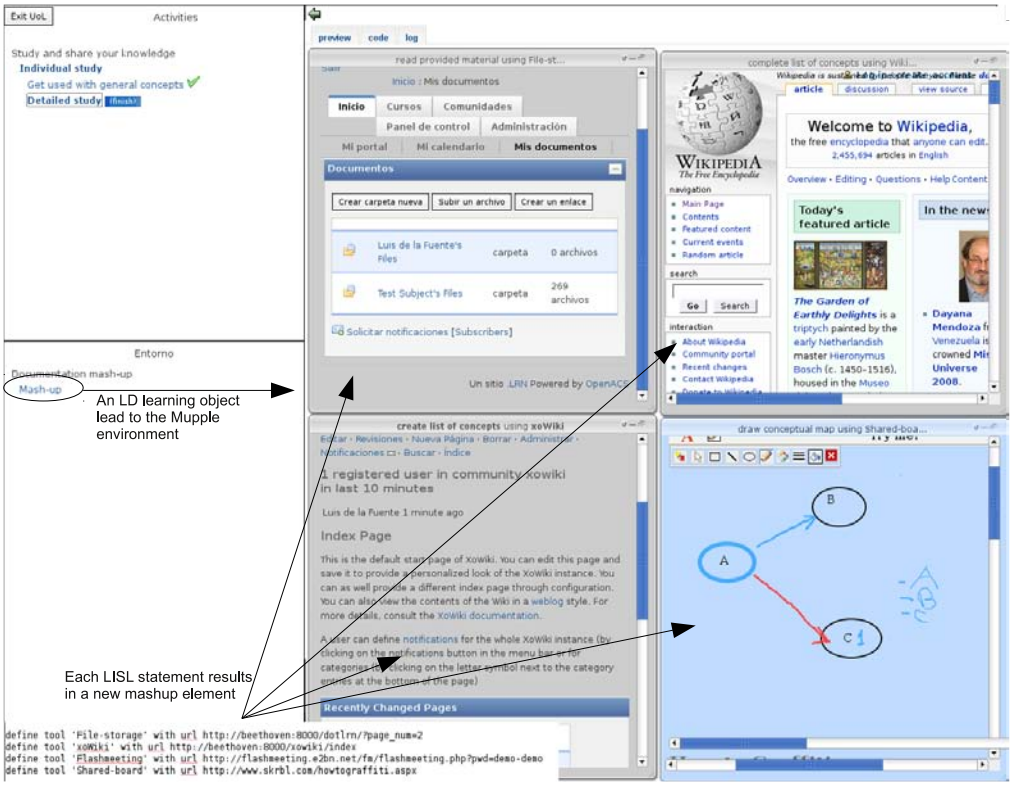


Figure 6.8: Resulting working environment when combining IMS LD and LISL.

In the UoL used in the example, the flow consists of two sessions, modelled as **acts** in

IMS LD. In the first session, learners have to study a set of initial documents and write a list of the main discussed concepts. Then, they have to explore more deeply these concepts and draw a conceptual map, graphically linking them. The activity starts with the reading of the documents available in the course repository. Then, students must use Wikipedia to clarify those concepts that not properly understood. Finally, a wiki tool allows writing the list of main concepts and a mind-mapper tool allows creating conceptual maps. The resulting structure is depicted in Figure 6.8.

Students access the activity environment and are able to modify it either modifying the source code, or using the graphical interface that allows to open, close, move and iconify each box. With this structure, an additional step (i.e. search a concept in Google), or the modification of one of the tools used by existing steps (i.e. one student prefers CmapTools for conceptual maps) can be easily achieved.

6.4.4 Discussion

The combination of IMS LD to define the learning flow (macro-script) and LISL to detail scaffolding details on activities (micro-scripts) may improve the runtime flexibility of *e-learning* courses. In this stage of the course, teachers have to be able to modify content, reconfigure services, and change termination conditions on an activity or its visibility. Students should have the possibility of adjusting the learning environment to their personal needs: select the tools to use and change their own profile.

The Learning Design specification offers a reasonable degree of flexibility at run-time. Level B properties and conditions can be used to modify the course behaviour even if it is already running. However, these changes must be explicitly stated during the authoring phase. Used to provide a framework where activities can be performed, the LISL language allows students to easily configure the environment and share preferences with peers.

The deployment of the example presented in this section required the ad-hoc implementation of LISL integration in GRAIL. Despite this task did not impose a high cost, the integration method might not scale to other situations with more fine-grained requirements. Two main conclusions can be extracted from the developed proof of concept:

- IMS LD provides good support to the macro-script orchestration of learning flows. However, it has problems dealing with fine grained details of particular activities. At this level, it is better to use case-specific tools rather than overscripting with IMS LD.
- The integration of case-specific tools is not natively supported in IMS LD. An ad hoc integration was implemented for the purpose of the presented example, but a more generalizable method would be required to make a reusable proposal.

These two conclusions (integration of case-specific tools and a generalizable integration method) were considered as requisites of the proposal presented in this dissertation.

6.5 Conclusions

This chapter presented the experiences enacted during the first step of the methodology: the characterization of the problem. Such experiences ranged from the theoretical discussions to courses enacted in real situations and covered the complete course life-cycle imposed by the use of IMS LD. The translation of an engineering course into the IMS LD vocabulary was completed for the analysis of the authoring phase. The deployment and enactment were studied in a course with participants geographically distributed and actively collaborating in the course activities. GRAIL was used in the deployment and enactment of such experiences,

and served also for the implementation of a proof of concept of the combination of IMS LD and mashups.

One of the motivations of this first phase of study was the absence of existing experiences documented in the literature. The publication of the developed experiences in different conferences [82, 115, 116], workshops [216, 215] and journals [208] evidences the relevance of the work and the interest of the research community in the field.

Several limitations regarding the use of IMS LD were detected in the developed experiences. A summary of the most relevant ones is presented as follows:

Difficult authoring phase. Despite there is a new generation of authoring tools relaxes the problem, there is still a need to understand the IMS LD model in order to create courses. In practice, this is an obstacle for practitioners to adopt the specification. Due to the existence of research projects [217, 218] oriented towards solving the authoring problem, such limitation is out of the scope of the work presented in this dissertation.

Lack of flexibility. The course authoring phase finishes when the UoL is exported to a zip file. Once the course has been exported, the inclusion of latter modifications in the course flow or content is no longer possible. However, unexpected situations appear in real courses and IMS LD does not provide mechanisms to handle them.

The data flow problem. Especially on collaborative courses, the output of one activity is sometimes used as the input of the following one. IMS LD does not provide methods to define such flow of artefacts. The flow modelling can be done with *properties*, but their use is time consuming and error prone.

Integration with external tools. Distance collaboration requires the support of tools that are usually accessed via Web. IMS LD does not provide support for such tools. The result is that their inclusion in a UoL translates into a loss of adaptive capabilities, replicability and usability of the course.

Uses of IMS LD. The purpose of the specification is to provide a mechanism to orchestrate learning activities with their related resources and services. This fact is sometimes forgotten by practitioners, who try to use IMS LD for complex tasks (e.g. calculations, group management) with poor results in terms of performance and functionality of the resulting UoL. This misunderstanding of the specification causes the authoring phase to be even more complex.

The work presented in this dissertation focuses on two aspects: first, the proposal and implementation of the required functionality to provide more flexibility to the course lifecycle (Chapter 4.3). Second, the proposal of extension of the specification that allows to integrate third-party tools in IMS LD courses (Chapter 5).

A lesson learned from the deployment of the presented experiences is the inherent difficulty of the replication process. On the one hand, the immature state of the technology puts a risk on the enactment of the learning flow and requires the researchers to be alert to unexpected events. On the other hand, it is not always feasible to arrange all the human resources required to deploy authentic experiences, so they have to be carefully prepared so that the risk of failure is reduced to the minimum.

Chapter 7

Evaluation of the proposed model

> If you don't crack the shell, you can't eat the nut.
>
> Persian proverb

7.1 Introduction

According to the methodology adopted in this dissertation, presented in Chapter 1.3, the deployment and latter analysis of experimental work is required for the validation of the presented proposal. Validating experiences are oriented towards observing to what extent the *Generic Service Integration* proposal solves the limitations detected in the literature and in observations of previous experiences. The validation phase iterated with the definition of the proposal so that the weaknessess highligthed by the first experiences served as feedback for the next iteration of the design process. As a result, the last presented experience was supported by a more robust system than in the first ones.

The experimentation performed in this dissertation was based on the deployment and analysis of different *cases of study* [219]. Each case of study is a world unto itself and the conclusions extracted are, strictly speaking, only applicable to the case under study. The generalization of the results depends on the particular circumstances of the scenario and lays on the subjectivity of the interpreter. Cases of study are applied in those scenarios where there is a high number of factors that may affect the results and it is difficult to recreate an experimental situation without these factors. This is exactly the particular case of learning experiences, where the results are influenced by (just to mention some of them) environmental, historical, personal and institutional factors. Besides, the few experience in the field make it difficult to perform another types of research, such as literature review or surveys. Thus, case study research is therefore applicable to the research presented in this dissertation.

The developed cases of study differ among themselves in several factors such as the number and profile of the participants. Each experience offered different data that enabled their analysis. Due to this heterogeneity, it is difficult to establish a common procedure for the data analysis and, as a result, each experience has been evaluated considering the particular needs of the enactment scenario. In all cases, qualitative data allowed to identify

the findings, while quantitative data was used to reinforce the conclusions. That is, the analysis of the experience followed a mixed evaluation method [11].

The developed experiences were held in a sequence that allowed the researcher to step-by-step analyse the proposal from its simpler fact to the more complex concerns. Following this paradigm, three cases of study were deployed, each of them built on the lessons learned from the analysis of the previous ones. The first of the presented experiences was oriented towards the analysis of the feasibility of the GSI courses when they are enacted by practitioners. The first objective was to determine if the replication process was realistic enough for large-scaled scenarios. The second goal was to observe how the users understand the model. The second experience aimed at the study of the support provided by GSI in the enactment of traditional learning methods such as *Project Based Learning* when applied to a large number of students. The impact of the technology-driven orchestration in the model was also under observation. Finally, the third experience analysed to what extent the proposal provides support for the deploy of innovative learning scenarios combining different technologies, spatial locations and activity types.

The rest of this chapter is organized as follows: Section 7.2 presents the results of a workshop with the participation of instructors from primary and secondary education with no previous knowledge of IMS LD. Next, Section 7.3 details the use of GSI for the support of a programming course with the participation of 425 students and 8 teachers. The last experience, presented in Section 7.4, is based on a previous scenario where the use of mobile phones and different spatial locations posed excessive workload to practitioners. The experience demonstrated that the use of GSI alleviated the required administrative workload. Finally, Section 7.5 presents the conclusions extracted from the experimental phase of the dissertation.

7.2 Feasibility of GSI driven adaptation in real scenarios

Education has undergone significant changes especially in the area of teaching strategies. The widespread use of information and communication technology allows teachers to access a larger variety of resources. Furthermore, students may use Internet as a vast catalogue where information can be searched and used to complete the activities proposed by the teacher. The variety of learning scenarios deriving from this change is breathtaking. As a consequence, students may take now a much more active role in educational experiences, and these experiences can be tailored to their preferences.

Ideally, instructors may prepare hypermedia material such that the resources offered to a student depend on her interests, profile, previous knowledge, past performance, or many other possible factors. This material is called *Adaptive Hypermedia* [220]. It is important to establish the difference between *adaptable* and *adaptive* content [221]. Adaptable material is meant to enable users to adapt themselves the content layout and navigation support to their preferences. On the contrary, in the case of adaptive material, these adjustments are performed automatically based on observations of the user behaviour. There are different types of adaptive learning schemes depending on the elements to modify [222]: Content adaptation, where the same learning activity is presented to all the students but with different material; flow adaptation, where a different set of activities is chosen for each student; and interface adaptation, where the same information is presented to all the students but with different layouts.

There are multiple aspects to consider when adapting the content of a learning experience. For example, Felder and Silverman [223] defined a set of learning styles as the basis to select different types of resources. In general, adaptation can be performed based on student

goals, preferences, background, personal interests, etc. [224]. Adaptation does not always depend exclusively on personal data. Learning experiences can also be changed based on the surrounding environment. For example, Muntean [225] studied how to adapt learning material based on the *Quality of Experience* of the used network, and Brown [226] studied the adaptation of an experience based on the learners' spatial location.

Another relevant factor in the adaptation of a learning experience is the delivery mode. A first approach is to use a dedicated platform providing a totally self-contained and fully functional adaptive learning environment to the users independently of any other tool. An example of this type of tool is *AHA* [227]. A different approach is to use a pedagogically neutral framework allowing the use of adaptive learning schemes as well as any other learning strategy. The IMS Learning Design specification (hence IMS LD) [203] falls into this category. IMS LD is a formalism conceived to create, deploy and enact interoperable, reusable and self-contained learning experiences. The specification allows the use of conditional expressions and properties to define multiple sequences of resources, and therefore, it is a suitable formalism to define and deliver adaptive learning material, as claimed by Burgos *et al.* [130]. A learning experience described in IMS LD is then enacted by a so called *engine*, a computer program that deploys the appropriate sequence of resources and environments to the participants.

But the *properties* used in a learning experience described with IMS LD can only be modified through expressions in the resources included in the course. In other words, the engine interpreting the choreography is self-contained, all properties are interpreted and modified within the engine. From this point of view, IMS LD can be defined as *closed to external properties*, understanding "external properties" as those managed outside the engine. The main consequence of this limitation is the impossibility for IMS LD courses to obtain information from external sources. That is, a learning experience cannot be defined, for example, to use conventional third-party Web 2.0 tools hosted in external platforms and adapt the environment depending on the activities that took place in these platforms. If, for example, a IMS LD learning experience uses an external collaborative virtual whiteboard, its use may be scheduled, but the adaptation cannot take into account any of the (potentially useful) events that occurred in this third-party tool.

Generic Service Integration, the proposal described in Chapter 5, allows the exchange of information among the IMS LD player and third-party tools. It is therefore a mean to design IMS LD based adaptive activities that rely their support on external tools. The feasibility of the proposed approach was evaluated in a pilot experience attended by 22 teachers from primary and secondary education with no previous knowledge of IMS LD. These teachers were told to manage a course containing adaptive material based on the results of a questionnaire hosted in Google Forms (see Chapter 5.4.1). The case of study was aimed to validate the GSI model and to evaluate the teachers' perception of the proposal. The experience used GRAIL [10] as supporting software. This engine includes a module implementing the exchange of information between the engine and generic external services as described in the GSI architecture.

The abovementioned experience is detailed in [202]. This section explains the details of the enactment of the case study: first, a description of the audience and the activities is given; then the methods to evaluate the experience are presented; finally, the conclusions obtained from the performed qualitative and quantitative analysis.

7.2.1 Description of the experience

Demographic data

The experience was deployed in the context of a series of science oriented workshops, where the presented one was advertised as *Computer support for the creation of adaptive content*. Considering the title and the topic of the workshop, the expected audience was teachers from K-12 or high schools interested on new methods of teaching and, more specifically, on using computers to support these methods.

Participants were volunteers that had to register days before workshop was held. A total of 22 persons registered for the experience: 17 women and 5 men. Among them, there were 9 teachers from K-12, 3 from high school and 6 were counselling psychologists. 38, 9% of them claimed to have advanced computer skills, while the rest had basic computer skills. Their computer skills averaged 3,38 on a scale of 1 to 5. Each participant was provided with a laptop to work individually in the activities. The participant profiles are summarized in Table 7.1.

Variable	**Number**	**%**
Men	5	22,73
Women	17	77,27
Primary Education Teachers	9	40,91
Secondary Education Teachers	3	13,64
Counselling Psychologists	6	27,27
Unknown occupation	4	18,18
Average computer skills	3,38 (out of 5)	
Standard deviation in computer skills	0,83	

Table 7.1: Summary of participant profiles

Workshop activities

The activities were carried out in a single four-hour session. The objective for participants was to understand a previously-created UoL, and create a new one based on a similar template. The team of tutors guided the participants through the activities in the workshop. That is, there was neither written description of the activities nor an automated orchestration system. The UoL under study contained a flow with three acts where the second one was adapted depending on a questionnaire obtained in the first act. The structure was quite similar to the *What is Greatness* UoL [92], but obtaining the data from a Google Spreadsheet-based form, using GSI. Each participant was given an independent user space in the platform. UoLs were already deployed and instantiated so the first required step was to subscribe the students to the course.

The workshop activities were divided as follows: first, researchers' team introduced the IMS LD framework to the audience, focusing on the adaptive capabilities of the specification. This explanation was followed by a demonstration of the example UoL, showing how the adaptation was implemented in the working case. Then, participants replicated the same sequence of activities in their own computers, assuming the corresponding roles of the UoL. They changed from the teacher role to the student one when the flow suggested so, both roles were being played in parallel. At this point, the participants interacted freely with the learning flow, asking for help when required.

In the second part of the workshop, the participants used GRAIL to create a new version of the adaptive flow: they started from a UoL where the flow was identical to the one in the previous activity, but with no content. They used the editing functionality offered by GRAIL to add content to the course. The objective of this second stage was to consolidate the acquired knowledge of the specification, seeing if the provided example would fit in their personal working environment and, if not, what new features would be required.

Before finishing, participants were invited to take part in a group discussion moderated by the researchers. The goal was for participants to reflect on the possibilities of adaptive schemes and the integration of third-party services in the course flow. This final discussion also provided feedback about the content of the workshop and its methodology.

7.2.2 Evaluation Methodology

Research questions

The workshop was intended to study two different aspects of the proposed architecture: from the technical point of view, the experience helped to evaluate if the GSI model really satisfies the requirements discussed in Section 5.5 (pedagogical neutrality, reusability, self-containment, adaptability, collaboration). In particular, the research questions about the architecture characteristics were:

RQ1) Can UoLs using GSI be reused in completely independent course instances?

RQ2) Is it feasible to deploy such a scenario at a reasonable cost?

The workshop also evaluated the acceptance of the model by the participants. In order to keep improving the proposed architecture and the provided features, it is important to determine how teachers understand the model, its perceived usefulness and the most valued features. The specific research questions in this area were:

RQ3) How difficult is for teachers to understand the "script based adaptation" paradigm?

RQ4) How teachers ponder the feasibility of IMS LD based adaptation?

RQ5) Does spreadsheet-based question management lower the adoption threshold of IMS LD based adaptation?

RQ6) Are Web 2.0 tools currently considered as teaching tools by instructors?

Data sources

The case of study offered several data sources: the surveys filled by participants, a final group discussion with the observations of the researchers and the instantiation process itself.

Participants received a total of three questionnaires, referred as *Q1*, *Q2* and *Q3*. First, they were contacted by e-mail prior to the workshop and asked to fill a questionnaire to determine the participants' profile. The second questionnaire was presented before the break between activities, and was aimed at capturing their first impression of the adaptive learning scheme. Finally, in *Q3* they expressed their conclusions about the feasibility of the learning method to be deployed in real scenarios. The questionnaires had a mixture of quantitative and qualitative questions and their analysis followed the principles of a mixed evaluation method [11]. A summary of the most relevant questions presented to the participants is given in Table 7.2.

When course activities had finished, participants were invited to discuss or comment what they found interesting in the workshop. A group of five participants took part in this

Id.	Question	Comment
Q1: before the workshop		
Q1.a	Computer skill of the participants	1=Low-Skilled 5=High-Skilled
Q1.b	Occupation of the participant	Open question
Q2: before the final activity		
Q2.a	Difficulty/benefit ratio of adaptation	1=Too difficult 5=High benefit
Q2.b	Comprehensibility of the example	1=Incomprehensible 5=Very comprehensible
Q2.c	Difficulty/benefit ratio of the example	1=Too difficult 5=High benefit
Q2.d	Comments about the adaptive script	Open question
Q2.e	Realistic in a real scenario?	1=Unrealistic 5=Very realistic
Q2.f	Useful in a real scenario?	1=Useless 5=Very useful
Q3: after the workshop		
Q3.a	Difficulty/benefit ratio of the example	1=Too difficult 5=High benefit
Q3.b	Feasible to be deployed in your working scenario?	1=Not feasible 5=Very feasible
Q3.c	Is the access to the spreadsheet easy to use?	1=Not easy 5=Very easy
Q3.d	Does the spreadsheet provide any added value?	1=No added value 5=High added value
Q3.e	What is the spreadsheet useful for?	Open question
Q3.f	Web 2.0 tools to include in an adaptive course	Open question

Table 7.2: Summary of the questionnaires used in the workshop.

discussion, which offered relevant reflections to better understand the participants' view of the used tools and methods. Researchers methodologically guide the discussion between an informal conversation interview and guided interview [228]. That is, there was an informal plan that guided the conversation and, depending on how talkative the participants were, the interview reminded more one of the other methods.

Finally, the issues of the instantiation and enactment process itself offered the researchers data to determine the technical feasibility of the model.

Analysis methodology

Considering that the above listed research questions can be divided in two groups (technical feasibility and participants' view of the model), they were analysed using different datasets and following different methodologies. First, technical feasibility was qualitatively evaluated by considering the observations of the instantiation process and the technical issues occurred during the workshop. Second, the analysis of the participants' view of the model followed the principles of a mixed evaluation method so that qualitative and quantitative techniques were in use. This approach allows expanding an understanding from one method to another, at the time that it allows triangulating data so that the evidences shown by one method can be confirmed or refuted by the other [228].

The questionnaires included numeric questions with a Likert-scaled response, used for qualitative analysis; and open-ended ones where participants freely expressed their opinions for qualitative analysis. The qualitative analysis also considered the final group discussion as a valid source of data.

The goal of the case study is to see how teachers understand the IMS LD courses, its adaptive capabilities and the connection of the framework with Web 2.0 systems. The aim is to reveal an overview of the GSI model's feasibility in current education, i.e., map the phenomenon. This is why qualitative analysis was used. In such type of analysis, the size of the sample is not an issue because the factors and/or opinions tend to repeat after a low number of questionnaires [229]. It goes without saying that there is no guarantee of identifying all factors that affect the phenomenon, but the most relevant ones are usually identified with such analysis. Since a case study is influenced by environmental factors and cultural issues, there is no room for the generalization of the results [219]. However, this is not an issue for the sake of this study because generalizing to the whole population of teachers is not a goal of the research. It is worth to mention that the statistical analysis described later suggests that some of the results could have been obtained also in larger samples. The worthwhile assertion here is that there is a chance of discussion, rather than asserting the generalization itself.

There is an intrinsic subjectivity on qualitative analysis. It could be argued that the results of the presented study are biased by different factors of the workshop, and that researchers have emphasized some factors more than others. To promote objective results as much as possible, the quantitative analysis is used to reinforce the findings of the qualitative one. The numeric questions included in the questionnaire allow extracting more objective conclusions than the text-based ones. In the context of this study, they were included to confirm the findings of qualitative analysis. To ensure that the mean and the standard deviation of a sample has not been randomly produced, statistical analysis allows detecting those situations in which the results are significant or not, even when the size of the sample is small [230]. In the presented study, quantitative data is represented by their mean and standard deviation. In those cases where two paired samples are compared, the T-test was used to examine the significance of the results.

7.2.3 Results

The instantiation process and the workshop itself serve as response for research questions *RQ1* and *RQ2*, formulated in Section 7.2.2. First, each participant interacted with a different course instance. All the proposed activities were successfully completed by the participants, which means that they could work with their individual course instance. Since there were no reported problems specifically related to the interaction among IMS LD and Google Forms, it can be said that GSI-shaped UoLs are as reusable as IMS LD courses are (RQ1). The deployment of the UoLs was performed by the researchers prior to the workshop, so the first participants' activity was to instantiate the courses. The process was repeated once per each participant, so the steps required for the course instantiation were repeated a total of 25 times. This task revealed to be highly repetitive and therefore easy to automate. The creation of an automation script reduced the time cost of administrative tasks so they remained into reasonable values (RQ2). Considering the deployment and enactment process it can be said that GSI-shaped UoLs maintain the self-containment and reusability of IMS LD.

The answers gathered from the participants were used to analyse the response to the rest of the research questions. We now analyse how participants perceived the usefulness of the architecture proposed in Chapter 5 of this dissertation, when applied to adaptive course material. The analysis is based on the 18 valid collected answers. A graphical representation of the obtained answers is shown in Figure 7.1, where the corresponding questions are presented in Table 7.2.

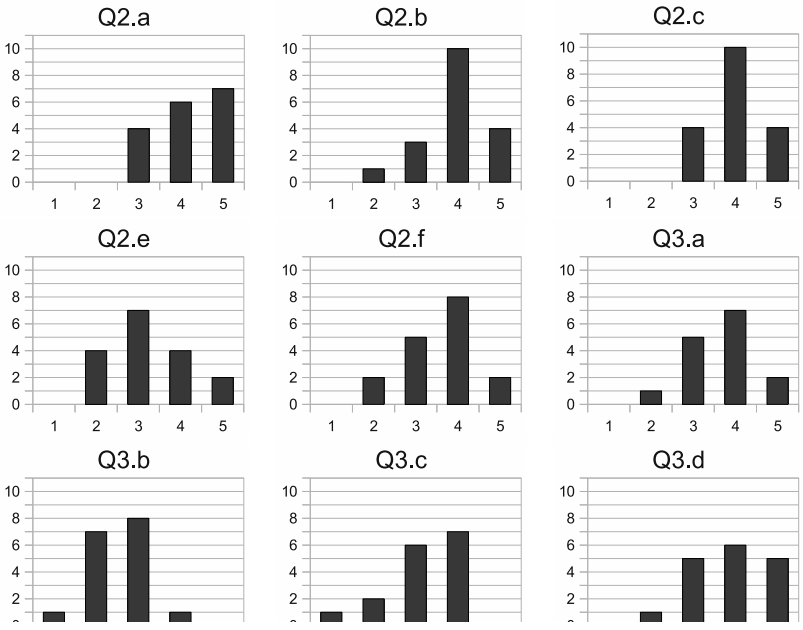


Figure 7.1: Histograms of the obtained answers. Y axes represents the number of responses, X axes is the option number.

Research question RQ3 regards how well the participants understood the script-based adaptation paradigm. The straightforward facts that support the comprehensibility of the approach are, first, that all participants successfully developed the proposed activities and,

second, that they expressed a positive opinion of the workshop. Examples of this positive opinion in the questionnaires are: "reasonable difficulty, and interesting", or "it seems easy to use and with a fast deployment for teachers with no experience with computers". The participants agreed on the argument that they had preferred to have more time to complete workshop activities, but the questionnaires do not contain negative comments on the comprehensibility. The perception of the researchers is that the model was quite well understood by participants. This perception is also supported by quantitative data, summarized in Table 7.3.

Question	Mean	Standard Deviation
Q2.a	4,18	0,81
Q2.b	3,94	0,8
Q2.c	4	0,69
Q3.a	3,67	0,82

Table 7.3: RQ3 relative questions.

The value obtained in question *Q2.b* shows that the example, the UoL with GSI, was quite comprehensible (3,98 out of 5). Considering the values obtained in questions *Q2.a* and *Q2.c*, it can be said that course participants perceived the potential of adaptation and use of third-party tools, while they felt optimistic about the difficulty of the implementation process. This perception on the difficulty of the creation process evolved after performing the final activity of the workshop: The mean value of the answer to *Q3.a* shows that the participants realized the difficulty in the creation of adaptive material. The t-test for the comparison of *Q2.b* and *Q3.a* samples ($H_0 : \mu_{Q2.c} = \mu_{Q3.a}$; $H_1 : \mu_{Q2.c} > \mu_{Q3.a}$) reveals that there is statistical significance of the result (p-value = 0.03408).

Half of the participants explicitly mentioned that the implementation of the UoL in an authentic situation would be beneficial for students' learning and that it would increase their motivation, at the same time that they recognize that teacher training would be mandatory to success in such implementation. The perceived feasibility of such type of learning material in a real situation is the matter of research question RQ4. Despite the positive view of the enacted example, they expressed their lack of confidence on its effective adoption. This view is supported by some remarks such as "It's nice, but it appears to be too abstract, and far from its application on specific material in the context of K-12 Education", or "I think it is very interesting, but maybe the difficulty appear while creating the material". Data shown in Table 7.4 reinforces this view: they considered the example to be fairly useful for a real scenario (*Q2.f*, 3,59 out of 5), but not feasible to be effectively deployed (*Q3.b*, 2,53 out of 5).

Question	Mean	Standard Deviation
Q2.a	4,18	0,81
Q2.c	4	0,69
Q2.e	3,24	0,97
Q2.f	3,59	0,87
Q3.a	3,67	0,82
Q3.b	2.53	0,72

Table 7.4: RQ4 relative questions

The previous observations confirm the results of other studies in the field of IMS LD: the most significant obstacle for the adoption of the specification resides in the complexity

of course authoring, rather than other problems like the understanding of the specification or the required training process of instructors [84].

While performing the workshop activities, participants' comments and questions were oriented towards understanding the process and the conditions imposed to the adaptation. The researchers' perception was that the participants did not make a clear distinction among the course flow manager and the spreadsheet tool. That is, they did not perceive the spreadsheet as something external. Regarding the complexity of the authoring phase, rules for questionnaire data management are easier to be edited with a specialized tool like a spreadsheet, rather than using IMS LD vocabulary. This is the conclusion extracted from Table 7.5, which provides and answer to research question RQ5. With no previous knowledge of the specification and no experience using IMS LD conditions, course participants had a positive opinion about the use of the spreadsheet. For example, "It's easy to analyse the received data" show the relevance of a good interface; "Store registers so the teacher can view students' progress, and maybe obtain a graphical representation" emphasizes the potential on the use of specialized tools. This positive perception of conditions management contrasted with other experiences on the use of IMS LD [231], where conditions authoring were considered error prone and difficult to understand.

Question	**Mean**	**Standard Deviation**
Q3.c	3,19	0,83
Q3.d	3,88	0,86

Table 7.5: RQ5 relative questions

To evaluate research question *RQ6*, the number and variety of tools proposed in answers to *Q3.f* were considered. If it is taken into account that the participants were volunteers interested in supporting teaching methods with computers, the researchers found surprising that only two of the participants proposed tools to incorporate in learning material: wiki, e-mail and blogs were mentioned. The prevailing opinion can be summarized with "Any tool is suitable for me, because I have so poor formation in the field that any tool is a novelty". Despite that educators perceived the potential of Web 2.0 tools in their learning context, they do not currently consider these tools to be included as part of their courses and the lack of training is perceived as one of the major determining factors.

7.2.4 Conclusions

The feasibility of the GSI proposal was evaluated in an experience consisted in a workshop with 22 participants with teaching background and basic computer skills. The workshop participants performed a practical activity aimed at understanding what IMS LD is and how GSI can be used to create adaptive activities. There were two main aspects to evaluate: the feasibility of course deployment (technical perspective) and the teachers' understanding of the model.

From the technical perspective, the results show that GSI does not impose any loose in the course replication process. This means that Units of Learning with GSI keep the adaptability, self-containment and reusability initially provided by IMS LD. The absence of remarkable problems during the deployment and the easy automation of the process allowed the quick creation of course replicas. Furthermore, the use of a specialized tool such a spreadsheet to manage the data provides the system with better usability.

Data gathered from the audience reveals that the main obstacle for IMS LD adoption resides in the authoring process, while teachers acknowledged the potential Web 2.0 tools in-

tegration. According to participants' opinion, the inclusion of spreadsheets to grade student responses helps on data analysis and to view student's progress.

7.3 Orchestration of Problem Based Learning

The experience presented in Section 7.2 demonstrated the replicability of the GSI paradigm and its suitability to be applied in real scenarios. The next step is to test the expressiveness of the proposal in real educational scenarios. That is, to check if GSI is able to model an already existing course for its later deployment and enactment. The enactment of a course with the same learning flow but with a different technology does not necessarily improve the course features nor solve the existing limitations. That is, a successful experience should provide an added value that cannot be provided by other methods.

This section presents an experience deployed during the 2009/2010 academic year in the University Carlos III of Madrid. The *guinea pig* was a Java programming course for freshmen students. Older editions of the course suffered from a scalability problem that did not allow deploying innovative (but time consuming) teaching strategies such as Project Based Learning. The result was a high dropout rate and an overall poor students' performance. The use of GSI was intended to promote the replicability of an automatically orchestrated learning strategy, so that it can be applied to a large number of students with a reasonable cost. For the purpose of the experience, a Project Based Learning strategy designed for a 50-students group was instantiated several times so that the orchestration system managed the learning flow of 425 students. The analysis presented in this section has been extracted from the content of [183].

With the course successfully deployed and enacted, the analysis of the experience is focused on the following topics: first, the impact of the method in the students' and teachers' workload. The former were supposed to work continuously, while the later should not devote too much time to management tasks so that they can focus their efforts on tutoring the projects. The second analysed aspect is the results obtained by the students in the course. The analysis aims at determining if the learning method had a real impact on students' performance. This second aspect is more related to with the use of PBL that to the technology used for the scafolding.

The rest of the section is organized as follows: the problem statement and a revision of the solutions found in the literature are given in Section 7.3.1. In Section 7.3.2, the delivery method used in the course is detailed. Then, Section 7.3.3 explains what data was used to evaluate the validity of the proposal and how this data was gathered. Evaluation results are presented in Section 7.3.4. Finally, Section 7.3.5 draws the conclusions of the presented work.

7.3.1 Problem statement and relevant literature

Programming courses are an important part of engineering curricula and an essential part of most engineering programs [232]. However, learning to write programs is recognized to be a challenging subject for students due to the required logical thinking and the implied abstract concepts [233]. Programming courses pose also a challenge to teachers because of the difficulty of grading students' code in a fair and time-efficient manner [234].

As a consequence of these difficulties programming courses suffer from high dropout rates, especially in the case of freshman students as documented in [235]. There are several factors that cause these poor results, but the most relevant are time and motivation [236]. Programming is completely new for most freshman students and they find the required concepts

difficult to understand. This fact affects their motivation and perceived self-efficacy, which is another relevant factor that is reported to cause the observed high dropout rates [237].

Project Based Learning (PBL) is a teaching technique that increases students' motivation. It is based on the constructivist theory, which promotes the active participation of learners in the learning process so that they are able to build their own knowledge. PBL is successfully used in programming courses, but scaffolding such a learning flow poses a great cost to teaching staff [238].

The application of the Declaration of Bologna in Spanish universities promotes a shift from a knowledge-based learning model to a competence-based one. The straightforward consequence is the need to apply teaching methods based on user practice, encouraging continuous work during the semester. Constructivist methodologies are therefore acquiring relevance in the development of the syllabus, where project-based learning activities are an appropriate choice.

The scenario considered in this work is a freshmen programming course with 425 students in a higher education institution. During the course, students worked in pairs to develop code for the solution of a complex problem. Students were divided in 12 groups and tutored by 8 teachers. In such scenario, all assignments require a scalable deployment and grading methodology. In previous editions of the course, the students submitted their code in a single final submission. This delivery method was required due to the impossibility for the teaching staff to receive and grade several submissions during the semester. The consequence of this method was that the students did not receive proper feedback until they finished their assignments, and the quality of the submitted solutions was lower than desired.

This section presents a GSI based method to scaffold a project-based learning strategy that promotes students' continuous work and reduces the workload of the teaching staff by automatically testing the submitted code. The activity descriptions were delivered to the students based on the results of the automated tests, so each pair of students worked at different, self-regulated paces. IMS LD was used to deliver the activity flow. The code testing was performed by a Web based delivery platform, a tool developed for this purpose. The linkage between the activity flow and code testing was mediated by GSI. The proposed delivery method was deployed in the second semester of the 2009/2010 course, as part of engineering programs at Carlos III University of Madrid. The results show that the new method was well received by students and the evaluation reveals a significant decrease in the number of dropouts in contrast to previous editions of the course.

Literature review

Grading the code submitted by students in programming courses is recognized to be a tedious and error-prone task [234]. The problem has received significant attention in the research community. In 1989 the TRY system was developed and applied in Computer Science curriculum [239]. The grade was assigned based on the results of test cases applied to the submitted code. The conclusions of this work claimed that the management of a large number of assignments performed by students is only possible thanks to the automation of the grading process. Similar conclusions are drawn by Feldman and Jones [240].

Cheang et al. [234] deeply analyses the factors involved in a fair grading system. According to this work, test cases are helpful to evaluate the correctness, robustness and efficiency, but are not suitable to determine the maintainability of the code. Thus, a human review must be still present. One of the drawbacks of code testing as a grading method is the need to reduce the assignment size, due to the difficulty to create test cases for large programs.

Project-based learning (PBL) techniques can be used to increase student motivation and reduce dropout rates [238]. With PBL, the achieved learning comes from the process of understanding and solving an open-ended problem in a collaborative manner. This method

helps students understand the whole problem. It is widely used in education, and there is extensive literature that describes the use of PBL techniques in programming courses [241, 242, 243].

Nuutila et al. [244] apply the methodology with a successful reduction of dropout rates. However, when they explored tutorless scenarios, they claimed that the process is difficult to be scaled to large groups. Köse [245] proposes to support PBL task assignment through a Web based platform were both teachers and students are guided by a task-flow manager. The automated flow management reduced teachers' workload and increased the scalability of the methodology, but was only applicable in the described context because of the case-specific functionalities of the platform. Following a similar approach, García-Robles et al. [114] proposed a method to handle the activity flow with the IMS LD specification. This method focuses also on Web usage and enhances reusability and interoperability of the designs. However, due to the intrinsic limitations of IMS LD, they used the specification features as a design procedure, but not as a delivery and enactment method.

The orchestration method used in the experience combines the two reviewed teaching paradigms in a single course: on the one hand, a project-based learning methodology was applied to 6 week long projects in order to increase students' motivation. These projects were orchestrated using IMS LD. On the other hand, the grading system was based on automated execution of test cases, with the aim of reducing the workload of teaching staff. The information exchange between the activity flow orchestrator (IMS LD runtime environment) and the Web based testing system was mediated by *Generic Service Integration* (GSI).

7.3.2 Orchestration of the Learning Flow

Context of the experience

The context in which the experience took place is a programming course in a higher education institution: University Carlos III of Madrid, in Spain. The course is placed in the second semester of the first year programme, and its content depends on a previous course of the first semester. The students took another programming course during the previous semester. As a result, students who did not pass the first term course usually leave the second semester course during the first weeks. This course is taught simultaneously in four different degrees, all of them with a telecommunications engineering background. The number of students is consequently large, and so is the number of tutors.

One of the goals of the course curriculum is the encouragement of continuous evaluation methods. The course grading policy is as follows:

- 5 five-minute tests performed during lectures (5%)
- 2 midterm examinations (20%)
- 2 software projects, based on Project Based Learning methodology (20%)
- The cooperation and work shown during the laboratory sessions (10%)
- A final examination at the laboratory (15%)
- A final paper based examination (30%)

The experience description given in this section focuses on the scaffolding method for the software projects. A discussion of the whole grading methodology is out of scope of this dissertation and it is mentioned here to contextualize the experience.

In previous editions of the course, the students worked in pairs (also referred as "teams") in the software projects, and they performed the tasks in an unsupervised environment, with

the possibility to ask for the help of tutors and having to deliver the code in a single final submission. The obtained results qualitatively showed that the project-based approach encouraged the students to study programming, but the method presented several flaws:

- Instead on performing a continuous work, the students solved the task just some days before the deadline.
- Several students could not face the accumulated work so they dropped the course.
- Several groups did not understand the task, and they did not ask the tutor.
- Other students did not ask the tutor because they thought they understood the task, but they did not.
- The submitted code was, in most cases, below the expectations. There were many submissions that did not fulfil the requirements, and even code that did not compile.

The aim of the proposed scaffolding method is to promote continuous work, thus reducing the dropout rate and increasing the quality of the submitted code without increasing the teachers' workload during the course.

Orchestration Details

The described experience was held in the 2009/2010 course, with a total of 425 enrolled students from 4 engineering degrees. For administrative reasons, students were divided in 12 groups ranging from 21 to 46 members. Each group was assigned a tutor, with a total of 8 teachers tutoring the PBL activities within the different groups. Groups' schedules were coordinated, but there were minor variations on each group due to calendar restrictions. That is, all the students performed the same sequence of tasks, but in different dates.

The grading system of the programming course was not modified with respect to the previous year, except for the scaffolding method in the two projects. The project description was divided in two parts: first, the overview of the software application to develop; second, the detailed description of each task. The overview included the functional requirements of the project and also the submission instructions, so that students knew the exact value of each portion of the code. The detailed project description was fragmented in three modules, which were submitted and graded separately. The modules were designed in such a way that one part depended on the previous one, so they had to be developed in order.

The students initially received the overview of the project and the description of the first task to perform. They could also access a Web based delivery system to submit their code. Immediately after the code was received, the submission system compiled it and ran the corresponding test cases. The feedback offered by the test cases was presented to the students, who instantly knew if their code was successful. Once the students' code passed the tests, they were allowed to access the description of the next task. This method guarantees that the activities are performed in the planned order.

The test case was provided to students only for the first task. They first tested their code in their own computer and, when the code was correct, they submitted it. In the remaining tasks, the students were responsible for the development of their own test cases before submitting.

To encourage continuous work, a deadline was set for each submission. Pairs who submitted their code before the deadline did not have to wait until this date to receive the next activity description, so that they got more time to perform the task. Pairs who did not submit their corrected code before the expected date could keep working in the task and resubmit it, but they were penalized with the impossibility of submitting the last part of the

project: penalized teams were not allowed to finish the project, but they were not prevented from developing code in the expected order. The activity flow for students and teachers is depicted in Figure 7.2.

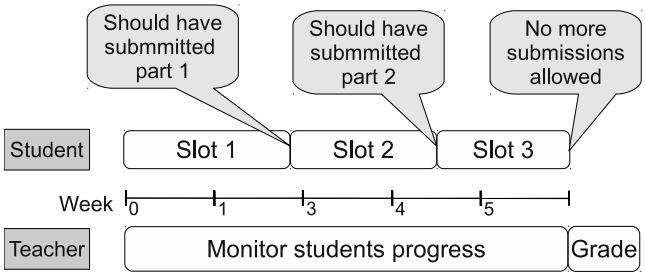


Figure 7.2: Timeline of the learning flow.

There were two laboratory sessions, one in the first week and another one half-way through the development process, dedicated to work in the project. In those sessions, teachers explained the software requisites, provided development guidelines and answered the most frequent questions. The rest of the time, students worked unsupervised and went to teachers' office hours if they got stalled.

The teachers graded the students' code after the last deadline and each part of the project was graded independently. The criteria considered correctness and maintainability of the code. The former was guaranteed by the testing system, so the teachers focused on maintainability. Incorrect code was also considered, but never graded above 5 (out to 10).

The students were warned of the use of plagiarism detection software and, at the end of the project, all submissions were analysed by sim [246]. The teaching staff individually interviewed the teams involved in the detected copies. The teachers then decided whether or not the case was an actual copy, and took the appropriate measures.

Technological Aspects

The IMS LD specification was used to capture the learning flow detailed in the previous subsection. The resulting Unit of Learning was deployed and enacted using GRAIL [10], which provides support for IMS LD and GSI. The remaining of this section is devoted to highlight the relevant parts of the design process[1].

The requirements of the learning flow were:

- The overview of the software to develop, as well as the grading method, must be always available.
- Activity descriptions are released only when the previous activity has been completed.
- An activity is completed when the submitted code passes the corresponding test.
- The students are working in pairs, so the completion of an activity by a single student must have the corresponding effect on the other team member.
- Each team should be able to work at its own pace. Teams delayed with respect to the schedule are penalized.

[1]The UoL can be downloaded from `https://gradient.it.uc3m.es/file-storage/view/pub/progsis-UoL.zip`

- Teachers are allowed to modify the initial schedule at any moment during the course.

The whole course was modelled as a single act with two activity structures. In the first of them, activities were delivered all at once (selection structure type), while in the second one the activities were sequenced in order (sequence structure type). The completion condition of these activities was a property to be set with a certain value.

The UoL defined one role per each team of students and a total of 26 roles (including the teacher role) were considered. No restriction was imposed in the number of students per role but, in practice, each role was populated by two students. As a result, each UoL instance was populated by a maximum of 50 students.

Conditions were established so that they considered the three time slots. The completion of an activity had different results depending on the slot in which it occurs. For example, the completion of the first task releases the second activity when it happens in the first slot, but nothing is released when it happens in the last slot (because of the penalization).

The code submitted by students was uploaded in a Web based delivery platform, a simple tool developed for this purpose that compiles the code and runs a test immediately after the student's file is uploaded. The delivery platform instantly prompts the result of the test, and also stores it in a log file along with the submitter identifier. This log file can be later accessed by simple HTTP requests.

The information exchange between the learning flow and the testing tool is performed by the GSI layer, whose major actions are two:

- From the user interface point of view, GSI retrieves the corresponding submission page of the delivery platform and modifies it so that it includes the identifier of the submitting student. Then, the page is presented to the student, to whom this process is transparent.

- From the point of view of the learning flow management, the GSI layer was scheduled to retrieve the log files each 10 minutes. The retrieved information is parsed so that the property 'activity-N-finished' is set to 1 if the student passed the test, and otherwise to 0.

The interaction among the three parts, IMS LD, GSI and the delivery platform, is depicted in Figure 7.3.

7.3.3 Evaluation Methodology

The deployment of the presented GSI based orchestration method aims to encourage students' daily work and study the effect on their performance. Therefore, the evaluation of the experience analyses the students' performance compared to what happened in previous editions of the course. Another aspect to be evaluated is the workload of the teaching staff, which should be kept into reasonable margins.

The data under analysis is obtained from three different sources, summarized in Table 7.6. First, the final course grades. The development of the software project is expected to have an impact over the rest of the graded tasks, so the analysis considers all the graded activities in courses 2008/2009 and 2009/2010.

The second data source is an anonymous survey that the students were encouraged to fill. The survey was anonymous and optional, and its completion did not affect the grade in any sense. The survey was available some days before the final examination, and was closed on this date. The survey was intended to capture the students' opinion of the scaffolding

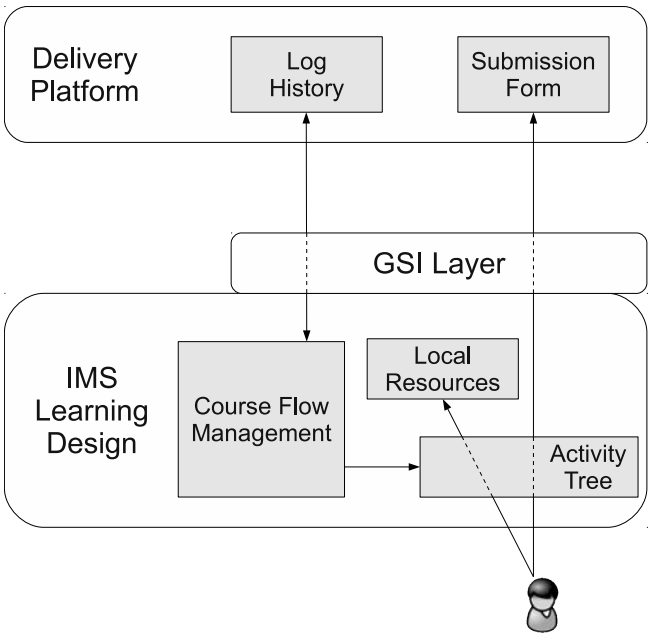


Figure 7.3: Architecture of the orchestration system.

Source	Description	Amount
Student's grades	Numeric data including the final grade and the partial results	425
Survey to students	Likert scale and free-text data. The answer was optional, anonymous and before the final exam	104
Survey to teachers	Likert scale and free-text data. Answers given after the final exam	8

Table 7.6: Data sources for the evaluation of the experience.

method. Some of the questions were free-text answered, some others were multiple choice questions, and the rest of them used a five-point Likert scale[2].

Finally, the teaching staff also filled a survey to qualitatively gather their opinion of the scaffolding method. The questions of that survey were both Likert scale and text based. The answers were obtained after the final examination was completed.

[2]A report of the survey results can be downloaded from `https://gradient.it.uc3m.es/file-storage/view/pub/progsis-survey.pdf` (in Spanish)

7.3.4 Results

Students' Workload

The presented scaffolding method is oriented towards promoting continuous work during the course. Considering the two projects, the students implemented and submitted a total of 6 tasks in a period of 9 weeks. Table 7.7 shows that 90.35% of the received submissions passed the test cases, and 92,8% of the functional submissions were received according to the expected schedule. The table also shows that 6,5% of the teams were not able to develop a correct solution in time, but they keep working and submitted it after the deadline.

The teachers were asked about their perception of students' continuous work and they answered 3.88 on average. Some opinions show that the students valued the relevance of the continuous work within the course. For example, one said that "there is no problem if you keep your work up to date", and another student stated that "this method forces you to finish the task on time, which helps people to keep tasks up to date".

		Project 1	**Project 2**	**Total**
Before deadline	pass	483	432	825
	do not pass	35	38	73
After deadline	pass	35	47	64
	do not pass	8	14	22
Total		543	441	984

Table 7.7: Submissions received

According to teachers, the difficulty and workload of the projects was similar to those proposed in previous years. This year, however, students reported an excessive time dedicated to the projects, in contrast to other parts of the course. According to the survey, 30,77% and 54,81% of the students worked more than 20 hours on each project (see Table 7.8). They recognized the projects as the most time-demanding task during the course, and they claimed for more weight of the projects in the grading system. Lot of students said that "we have really worked a lot hours in the project" and that "the projects stole you too much time, this is not according to their weight". This opinion is shared by the teachers, who said that "two projects means too much work, we should consider having only one project in the future". The interesting fact about this opinion arises when the course is compared with previous editions, where teachers did not perceived such excessive workload. It can be argued that the scaffolding method really forces the students to work, so teachers viewed the actual workload of the project.

The workload imposed to the students was also reflected in the intensive use they did of teachers' consulting hours. In general, students asked about the project more than in previous years, both in laboratory sessions and during consulting hours.

Despite the workload reported by students, the scaffolding method was considered beneficial. According to Table 7.9, they would like to use the system in future courses. Improvements to the system was suggested in the survey, the most demanded one can be summarized with this opinion: "I would add the possibility of repeating the submission once the code has passed the test, so that we can improve the code and its comments".

		Project 1	Project 2
Time spent	< 10 hours	13.46%	8.65%
	10-15 hours	37.50%	15.38%
	15-20 hours	18.27%	21.15%
	20-25 hours	17.31%	19.23%
	> 25 hours	13.46%	35.58%

Table 7.8: Time used by students in project development

Yes	69	66.35%
No	16	15.38%
N/A	19	18.27%

Table 7.9: Answers to the question: Would you like to use the same submissions system in future courses?

Teachers' Workload

The tasks required from the teaching staff were the following: first, they had to explain to the students the methodology of the project and the requisites of the code. Then, teachers tutored the students when requested, both in laboratory sessions and during office hours. Teachers were also responsible of setting the submission dates, so they had to be coordinated with teachers from other groups. Finally, they had to grade all submissions. Apart from setting the submission dates, these are essentially the same tasks than in previous years. That is, the scaffolding method did not increase the number of teachers' tasks. On the other hand, the teachers' workload required to perform these tasks changed in respect to previous years:

Office hours. As explained above, students asked more questions about the project than in previous years. Some teachers have used more time to answer these questions than the time they scheduled for the task. There was an agreement that attending students during office hours was the most time demanding task this year: "the semiautomatic grading system has increased the workload related to doubts resolution".

Grade submissions. This year, all the code graded by teachers had successfully passed the tests filter, so correctness was guaranteed. As a result, the grading process was limited to check for code maintainability, which is simpler than grading functionality. As one teacher commented, "the pre-grading has removed a lot of unnecessary work".

The management of submission dates required the coordination of the 12 groups, which was not always possible due to calendar restrictions. The consequence was that the dates were occasionally too close to the midterm exams. The Unit of Learning allowed modifying the dates, adding certain degree of flexibility to the system, but the more tasks imposed to students, the more challenging dates selection is.

One of the most demanding administrative tasks was to register the students in the IMS Learning Design runtime engine. Students had to create the teams among themselves and

report their decision by certain date. There were several teams reported past the deadline, and numerous team adjustments needed to be made during the project execution. IMS Learning Design does not offer the desired flexibility to accommodate these changes which translated in a peak in administrative tasks.

Drop Out Rate

A relevant fact extracted from the data presented in Table 7.10 is a significant decrease in the dropout rate: 36,23% of the registered students left the course before the final exam, in contrast to the 51,71% observed in the previous year. There are several reasons that explain this decrease. Among other factors, the project scaffolding has contributed to it. First, the increasing difficulty: the activity flow starts with the easiest tasks and the difficulty is increased gradually with each activity. When students finish the first activity, they instantly know that they will get a high percentage of this task's score, so they get engaged with the project. On the other hand, they instantly knew if they succeeded in their development, with a consequent increase of their self-efficacy. When asked if they perceived the scaffolding method as beneficial, the prevailing opinion was that "it is an incentive, because it is satisfactory to know quickly if your code works", and also "having to clear levels is like an incentive for our motivation".

Overall Course Results

The continuous work is expected to have an impact over the overall course results. As discussed above, the most straightforward observation is the reduction on the number of dropouts. There are also other interesting results extracted from the data presented in Table 7.10.

The scores obtained in the projects were significantly greater in the 2009/2010 edition of the course. As one of the students said, "once submitted, we know whether ours code succeeds or not". The immediate result is that the students were able to iterate until they got the right solution: the more solutions you get, the higher the grade. In the 2008/2009 edition the students only if the submitted solution was correct when the final grades were published, so there was no chance to improve the solution.

This increase in the project performance was expected to be reflected on the examination results. But, surprisingly, the results were worse than the previous year. The fact could be explained by the less time the students had to prepare the theoretical part of the midterm exams. For example, one student argued that "sometimes it seems impossible to study, because of the lack of time". Another possible argument to explain the midterm results is that the students "played" with the distributed grading system. That is, there were several project tasks contributing to the final score, so their interest on the midterm exams decreased in favour of the projects. Another plausible explanation is that students acquired better programming skills, but they worse theoretical skills. However, this argument contradicts the increase in results obtained in the final exam.

The course overall score was a bit higher than in the previous year, but the difference is not statistically significant (p-value$>$ 0.05). That is, those who took the course had an overall performance similar to the obtained in the previous year. There were more students who took the course, so there were more students who passed the course.

In summary, the proposed scaffolding method emphasizes the acquisition of programming skills, instead of promoting the knowledge-based learning. The overall results are similar in terms of the number of students who passed the course, but differ in the number of dropouts and the type of learning they acquired.

		2008/2009	2009/2010	p-value
Drop out rate		212/410 51.71%	154/425 36.23%	$3.0 \cdot 10^{-6}$
Project 1 score	mean: std:	4.76 4.00	6.63 3.38	$2.8 \cdot 10^{-11}$
Project 2 score	mean: std:	4.03 4.00	6.65 3.06	$2.2 \cdot 10^{-16}$
Midterm 1 score	mean: std:	4.80 2.15	4.28 2.31	0.0036
Midterm 2 score	mean: std:	5.00 2.53	4.14 2.26	$3.8 \cdot 10^{-5}$
Final exam score	mean: std:	4.06 2.54	4.54 2.24	0.0152
Total score	mean: std:	5.18 2.03	5.413 1.81	0.0965
Passed the course (over registered)		123/410 30.00%	180/425 42.35%	$9.7 \cdot 10^{-5}$
Passed the course (over non dropouts)		123/198 62.12%	180/271 66.42%	0.1697

Table 7.10: Student results in different course editions

Plagiarism

The data extracted from the plagiarism software (executed at the end of each project), is summarized in Table 7.11. There was a large amount of copied submissions, especially on the first project. The interviews with the involved teams revealed four main cases:

Shared work. Some teams joined their efforts and shared their code, even without changing it for the submissions. They argued that this was not plagiarism but collaboration. These groups were penalized because the aim of the projects was to work in pairs, and the task was designed for two-member teams, so each team must write they own code.

Allowed copy. Several teams let their colleagues to copy their code. They argued that they were just helping a friend. Both the team who copied and the team who allowed the copy were penalized.

Stolen code. There were teams that accessed others' code without permission. They did it, for example, looking in the trash of a shared computer. When one of such cases was proven, the producers of the original code were not penalized.

Unrecognised cases. There were few cases where the teams did not recognize the copy and the code was not similar enough (human reviewed) to assume copy. In those cases, there was no penalization.

In previous editions of the course, the plagiarism software was not used. Thus, the only observed copies were among teams of the same group because they were graded by the same

teacher. However, there was no mean to analyse copies from other groups. The analysis of the experience's data reveals the inter-group copies as the major source of plagiarism cases.

	Project 1	**Project 2**
Total teams	192	176
Teams involved in plagiarism cases	36	8
Penalized teams	29	4

Table 7.11: Copied submissions detected by the plagiarism software

There is an obvious decrease in the percentage of copies in the second project. It seems that the students realized the situation, so they preferred not to risk their work. It could also be argued that the students learned to better hide copied code, but it is unlikely if we consider the number of teams that started the project without finishing it.

The same effect of plagiarism on automatic grading systems was also reported by Cheang et al. [234]: first, students believe they can cheat the system and, after the penalization, they realize it is not worth the risk. It seems therefore essential not to delay the penalization, so more students will decline to copy in later tasks.

Additional Observations

Although not a priority in the project, students were expected to learn how to develop test cases: the first task had the corresponding test as an associated resource, but in the remaining tasks students were suppose to create such tests. In practice, students used the delivery platform as the only test mechanism and, in general, they did not create their own test cases.

The delivery platform performed quite reasonably during the course but, in very specific moments, there were performance problems caused by submissions containing infinite loops or code with very poor efficiency. The continuous work relaxed this problem: there was almost no "deadline effect" because each team worked at their own pace. The commented problems appeared especially at the end of the course, were the students' workload gets bigger and they find more difficult to submit in the expected dates.

7.3.5 Conclusions

This section presented an orchestration method to deploy and enact project-based learning activities within large groups without a severe impact on teachers' workload. The methodology is based on the use of IMS LD and GSI, which have been used in the design and delivery of the activity flow: IMS LD orchestrated the activity flow and GSI provided communication with a tool specialized on software testing. The method was applied in a programming course, but can be generalized to be used in any other discipline, with the requisite of having a semi-automatic method to grade students' submissions.

The students have worked continuously in the projects: instead of waiting until the deadline to submit the task (which is the common behaviour with no orchestration method), with the proposed approach they developed code every week in the semester. The analysis of the results shows a significant decrease in the number of dropouts. Consequently, more students than in previous editions passed the course. The automation of the task flow used for the project-based learning activities was well-considered by students, and the majority of them said they would like to use the same method in future courses. This system emphasizes

the relevance of programming skills over the theoretical knowledge of the subject, helping in the transition toward a competence-based learning method.

The impact of the method on teachers' workload was low and students were able to advance in the activity flow without the tutor intervention. The continuous work has encouraged student participation in the course and teachers reported an increase in questions in the laboratory sessions, as well as more frequent use of office hours than in previous years. With this participation, the teachers have obtained feedback from the students, so that they have realized the actual workload imposed by the project. Future editions of the course have to revise its size and schedule.

Apart from pedagogical achievements of the proposed method, this experience shows how the combined use of IMS LD and GSI allows teachers for the scaffolding of complex methodologies, able to be used in large student groups. The method has shown a high level of replicability, which means that the marginal cost of orchestrating a new group is low. Furthermore, the design of the first project was reused in the second one, demonstrating the reusability of IMS LD courses. Experiences deployed prior to the definition of GSI (see Chapter 6) showed that the lack of integration of the overall system was perceived by course participants as a usability problem, at the time that imposed severe restrictions on the reusability of the model. The major consequence of this lack of integration was that the teachers refused to deploy revised versions of the course arguing that such workload was not assumable. This experience shows how GSI highly increases the reusability and replicability of the orchestration method, at the time that the participants' perception of integration is no longer identified as a deficiency in the model.

7.4 GSI to increase the scalability of complex experiences

Available technology enables new learning scenarios where the learners play a more active role and the content can be adapted to their needs. The use of mobile phones is a good example of these innovative scenarios: interaction mediated by sort range communication technologies or the use of augmented reality are still to explore in the field of learning. These scenarios where different activities are mixed with different spatial locations and collaborative activities are called Computer Supported Collaborative Blended Learning (CSCBL) scenarios. One of their characteristics is that they pose a challenge to practitioners due to their high administrative requirements.

Such high workload is, in practice, a limitation for the execution of this type of scenarios. It was one of the lessons learning from an experienced enacted in the Universitat Pompeu Fabra, where the use of mobile phones in an outdoors activity was combined with in-class collaborative work. The experience showed promising pedagogical results, but the teachers were reluctant to participate in another edition of the learning flow because of the high administrative workload.

The use of GSI to reduce the workload of learning flows was already explored in Section 7.3. This experience has a relevant difference with the CSCBL case: the workload of the experience in Section 7.3 was related to teaching tasks (i.e. grade the students' submissions), while the workload of the CSCBL case regards the administrative management of the activities.

The challenge for GSI was therefore to determine if an orchestration method based on IMS LD and GSI is able to alleviate the limitations of the CSCBL scenarios. With this goal, the previous experience in the UPF was used as a case of study so it was analysed for the consequent proposal of an automated orchestration method. The theoretical proposal of such system is the matter of [182].

This section is devoted to explain the experience and its results. First, the manually orchestrated experience is presented, so that the reader can be familiar with the CSCBL learning flow. Then, the proposed orchestration method is detailed. Finally, the evaluation of the proposal consists in the enactment of the CSCBL flow within the proposed system.

7.4.1 A non-scalable learning flow

The Universitat Pompeu Fabra, in collaboration with the University Carlos III of Madrid, deployed a learning experience in the first day of the 2009/2010 academic year. The experience, called *Meeting the campus together*, was oriented toward freshmen students, who were in their very first day of their university years. One of the mandatory courses to these students is *Introduction to Information and Communication Technologies* (IICT), in which one of the aims is to give a global vision of the University and its resources.

In such context, the experience was oriented towards helping students in their first contact with the campus. The activity flow consisted of a mixture of individual and group tasks, performed in different spatial locations, which allowed the participants to interact with the campus resources and reflect about them. The participation on the experience was optional for the students. This subsection summarizes some elements of the experience. A complete analysis of this experience is provided in [247].

Description of the activities

The major goal of the experience is to allow the students to get used with the campus resources. The proposed sequence of activities was based on the Jigsaw collaborative pattern and therefore the collaboration among peers play a relevant role in the performance of the activities. This is because another objective of the experience was to help students meeting each other. The learning flow can be divided in three main phases: *explore the campus*, *explain the campus* and *reflect about the campus*. The three phases, depicted in Figure 7.4, are described as follows:

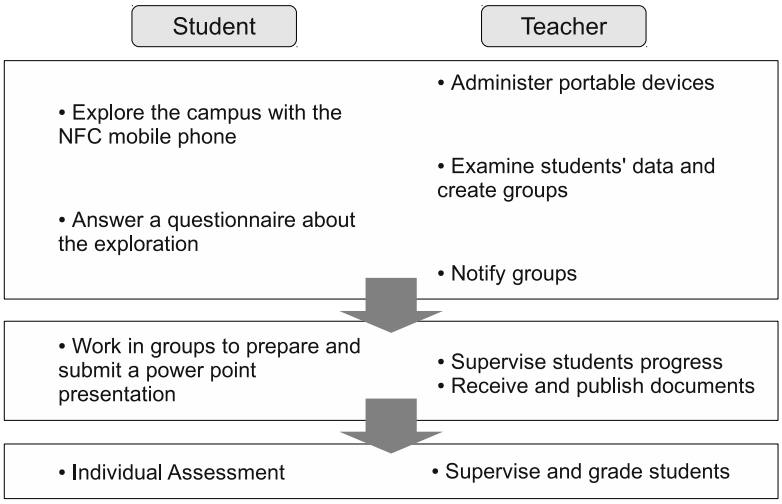


Figure 7.4: Learning flow enacted in the original experience.

In the first phase, *explore the campus*, the students were provided with NFC capable

mobile phones (Nokia 6131 and Nokia 6212) where NFCPlayer [248] had been installed. Then, the students explored the 5 buildings in the campus searching for the 50 NFC tags previously emplaced along the campus. When the students touched one of such tags with their mobile, it reproduced multimedia content related with the location (i.e. the building) where the student was. They had to complete the exploration in 30 minutes, so that they had not enough time to visit all the buildings and they had to choose their preferred one. Thus, they became *experts* of a particular building so they could be grouped in the following phase. After their exploration, the students answered a questionnaire with questions aimed at determining how much knowledge they had acquired from the activity.

During the exploration, the students the generated some data (log files in the mobile phone and the answers to the questionnaire) that allowed the teaching staff to classify the students according their level of expertise on the different buildings. Thus, they created five-member groups where all the peers were experts in the same building.

Working within the groups created by the teaching staff, the *explain the campus* phase consisted in a collaborative activity where the each group had to create a template based document and summarize the major aspects of their assigned building: location, resources, characteristics, etc. The created document was then uploaded to the used virtual learning environment (Moodle, in this case). The teachers collected all these documents and published them so that they could be accessed by all the students in the group.

The published documents supported the students in the third phase, *reflect about the campus*, where they had to answer a quiz with questions regarding the 5 buildings in the campus. The students used their own experience to answer the questions related to the building they had visited, and the documents created by their colleagues to find the answer of the rest of the questions. Some of the questions of the final quiz used QTI integration with Google Maps [249]. For these questions, students locate their answer in a Google Maps map and the system validated its correctness.

Some elements of the learning flow deserve to be underlined due to their impact in the organizational issues of the experience. The most relevant are:

NFC interaction. An innovative use of this technology was presented in the experience. The experimental use of mobile phones in learning activities required an administrative effort related to gathering and classifying the data from all the hardware devices.

Different spatial locations. The *explore the campus* activity was performed outdoors, the *explain the campus* was developed at home and the final assessment was done in the classroom. The experience aimed at exploring the impact of such combination of spatial locations in the students' learning.

Dynamic groups. The group activities were held in the second phase of the learning flow, and the groups were formed depending on the data gathered from the first activity. The impact of such administrative requirement in the course performance was also a matter of research in the experience.

Benefits and Limitations found

The experience was successfully deployed and enacted. The combination of different technologies and spatial locations revealed some benefits of the learning scheme. First, it was very motivating for students. Despite being optional, the participation exceeded the teachers' expectations and the students said that the innovative use of technology had encouraged them to take part in the course.

However, the most valued activity by the students was not the mobile based exploration, but the collaborative creation of a document. That is, the experience promoted the collab-

oration among peers, which is one of the transversal skills that they should master during their studies and their working life.

On the negative side, the administration of the experience was manually conducted by the teaching staff and imposed a considerable workload that was said to be error prone and time consuming. The analysis of students data, required to create the groups, took several hours to be accomplished. As a result, the experience could not be enacted in a single day and, for scheduling reasons, it lasted a total of two weeks. The experience was held in the first week of the academic year, when it is very common to find that the students drop out or change their studies. Thus, there was an unexpectedly high drop out rate in the experience, which caused the groups to be recalculated and thus increasing the teachers' administrative workload.

The teachers' workload increases with the number of participants. In practice, the learning flow was not applicable for a large number of students. Even with few students, the teachers said that the administrative workload was too high so they could not focus their efforts on tutoring, and they were reluctant to participate in future editions of the experience.

In summary, it can be said that the experience showed promising results but its lack of scalability and replicability posed a difficult obstacle to overcome.

7.4.2 GSI based orchestration method

The limitations of the experience presented in Section 7.4.1 were considered in the design of a solution based on the automatic orchestration of the described learning flow. The challenge was to create a system that integrates the gathering and analysis of the students' data, so that the workload of groups' formation is reduced. This section presents the solution that was proposed with the use of IMS LD and GSI.

Functional requisites

Considering the limitations identified in the previous edition of the course, an orchestration method for the learning flow should accomplish the following requisites:

Replicability. The workload imposed by the deployment and enactment of new course replicas should be kept into reasonable margins, so that the benefits of the learning flow compensate the administrative workload.

Scalability. The method should be able to be applied to a large number of students. Typically, to all freshmen students in a given degree.

Adaptability. Activities should be delivered to the students depending on the group they belong to. Thus, the orchestration method should have adaptive capabilities.

Flexibility. The method should provide guidance to practitioners and students so that they can follow the learning flow without difficulty. However, this guidance should be flexible enough to manage unexpected situations that usually happen in the course enactment.

Technological aspects of proposed orchestration method

The solution proposed to overcome the limitations of the first edition of the *meeting the campus together* experience is a script based orchestration method that automates the time consuming administrative tasks of the workflow. The proposed solution uses the following technology:

IMS LD is used to manage the activity flow. Thus, the course participants have to login in the used player so they can access to their corresponding activity description. The adaptive capabilities offered by IMS LD allow to dynamically create and dissolve groups, and to present the information to the course participants depending on the group they belong to. Also, *properties* provide a mean to manage the data flow while the features of GRAIL, the used IMS LD player, provide a flexible framework that allows the management of unexpected situations.

IMS LD has been criticized because of the difficulty of managing conditions. Dynamic groups' formation requires the creation of rules that are, in some cases, very difficult to express with IMS LD conditions and, in some other cases, simply not possible. For example, it is quite complicated to assign a group number to a student considering his/her available data, but it is not possible to create a condition that considers peers' information in the group assignment process.

One solution is to delegate group formation to a more capable tool. Thus, the proposed system relies on a spreadsheet for the grouping task. The used tool was Google Spreadsheet. Thus, the teacher uses a spreadsheet stored in his/her Google user space and filled with the data gathered from all the course participants.

The groups calculated in the spreadsheet are used by the IMS LD player to adapt the content delivered to the students. The exchange of information between these two systems is mediated by GSI, whose *GSpread* service adapter mediates the communication as described in Chapter 5.4.1.

The proposed Unit of Learning

The first step to reach the orchestration is to the translation the learning flow into the IMS LD vocabulary. Such formalization involves two main aspects: first, the definition of the activity flow in terms of the IMS LD vocabulary; second, the use of GSI tools to establish how and when the information is exchanged between the IMS LD server and the spreadsheet.

The IMS LD flow is composed by three acts, which correspond to the three phases that the course consists of. Three roles (two students and one teacher) are used to model the different course participants, where the only difference among the two student roles is the order in which they follow the activities of the first phase: students of type A starts with the Web exploration; mobile exploration is the first activity of students with type B. Each student decides when to finish the first activity and continue with the second one. That is, the completion condition is *user-choice*. The first act is completed when the teacher completes his/her *role-part*.

In the second act, *explain the campus*, each student access one of the five available activity descriptions. Which activity is accessed is determined by the corresponding *conditions* imposed to the so called *group_number* property. That is, the value of such property guides the student to one or another activity. The name of all group members is stored in another property, which is shown to the students in order to inform who their teammates are. At the end of the second act's activity, each group uses the output document they produced as the value of the corresponding role-scoped property. Thus, all submissions are stored with a regular structure and can be easily reused in the last act. The second act finishes when the teacher completes his/her *role-part*.

In the third act, the students access *QTIMaps* so that they complete the final assessment. To help them in their reflection process, they can access to other groups' documents, which is modelled as an *imsldcontent* document that simply show the properties whose content were populated by the groups' output in the previous phase.

Teacher's tasks modelling is much simpler than in the case of the students: each act contains a single activity with the description of what the students are doing and what the

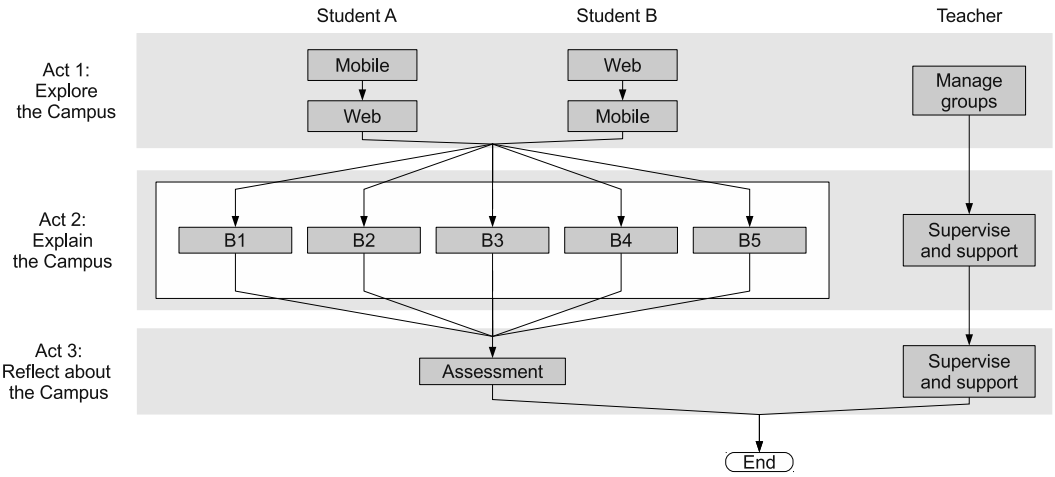


Figure 7.5: Diagram of the activity sequence expressed in IMS LD.

teacher can do to support them. The completion condition for all teachers' activities is *user-choice* and their completion causes the wrapping act to be set as finished. Figure 7.5 depicts the IMS LD formalization of the activity sequence.

The information exchange between the spreadsheet and the IMS LD player is triggered by the teacher's activity: when the groups have been created in the spreadsheet, the teacher sets his/her first act's activity (and therefore the act itself) as finished. This event causes the data request so that the information in the spreadsheet is used to fill students' *group_number*. Once all students have been assigned to a group, then the system delivers the second act's activity.

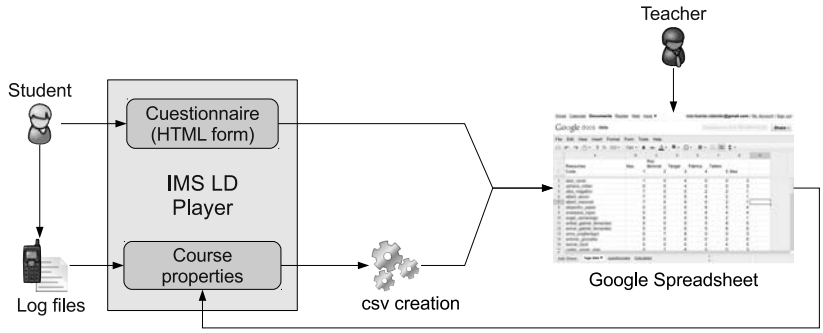


Figure 7.6: Data flow in the proposed system.

The information generated during the activities is evaluated by the orchestration system in order to create the groups and deliver the proper activity descriptions. It is required a method to fed the spreadsheet with the students' data. It includes the results of the NFC Player logs processing and the answers to the questionnaire. According to the functionality of *GSpread* described in Chapter 5.4.1, the answers to the questionnaire are automatically stored in the spreadsheet (one student per row) and it is known what answer corresponds to what student. To store the mobile phone logs in the spreadsheet, the following steps are

given:

1. When they finish the campus exploration, the students use a form to upload the generated log file. The log is stored as a IMS LD property so that all logs are regularly structured.

2. When the students have finished uploading their logs, the teacher downloads all of them to his/her own computer.

3. The regular structure of the logs allows programmatically process them. A script developed for the case is applied to the log files. The output is a *csv* file relates the students with the building that, according to the touched tags, they have visited.

4. The *csv* file is uploaded to the spreadsheet.

After that, all the data is in the spreadsheet and the teacher can create the formulae to create the groups with the case based restrictions. The data flow is depicted in Figure 7.6.

Differences with the previous course edition

However, this orchestration method imposes additional needs that did not appear in the original flow. The consequence is that the IMS LD orchestrated experience presents some differences in respect to the original learning flow. These differences are summarized in Table 7.12.

	2009	**2010**
Duration	The learning flow lasted several days	Two-hours learning flow
Physical space	Some of the activities were not performed in the classroom	Face-to-face activities
Mobile exploration	The students formed groups to explore the campus	Individual exploration of the campus
Course structure	All the students in the same course instance	Sessions of 25 students

Table 7.12: Differences between the IMS LD orchestrated flow and its original version.

The main restriction imposed by IMS LD is the need of computers to deliver the activity descriptions. In practice, it means that the students must go to the laboratory to perform the activities instead of being in any other place such as the library, or home. This fact limits the number of students that can participate in the experience. On the positive side, the orchestration method allows the experience to be quickly enacted, so several course instances can be held in the same day. In practice, the analysis of the experience showed that it is realistic to provide support for 100 students in a single day.

Another difference that was not caused by the technological support, but also impacted the course enactment is the optional character of the course: in the previous edition, the participation in the *Meeting the campus* experience was optional and rewarded with some points in the final grade. For administrative reasons, it was not possible in the IMS LD orchestrated edition, so the students gained no benefit (except the acquired learning and the colleagues they met) from their participation.

7.4.3 Details of the enacted experience

The IMS LD based orchestration method described in this section was enacted in the second day of the academic year 2010/2011. This experience, enacted with GRAIL, was aimed to evaluate the feasibility of the proposed orchestration method. The following details apply to the evaluation experience:

The participation in the experience was optional and was not rewarded with extra points in the final grade. Considering the number of participants of the previous edition, 80 or 100 participants were expected to register in the course. The performed deployment provided support for 100 students. The course flow was designed for 25 students (5 groups of 5 students each). Thus, the learning flow was replicated four times in the same day. The actual number of participants is presented in Table 7.13. It can be seen that the actual number was under the researchers' expectations. The most likely reason is the bad weather conditions: it is reasonable to think that students did not want to walk through the campus while it was raining.

Instance Number	**Timetable**	**Number of participants**
1	10:00 - 12:00	9
2	12:00 - 14:00	16
3	14:00 - 16:00	3
4	18:00 - 20:00	3

Table 7.13: Participants in each course instance.

Each replica of the learning flow was tutored by a different teacher. They were members of the IICT teaching staff and had no previous knowledge of GRAIL, IMS LD or GSI. The teachers tutored the learning activities and were supported by the researchers in the development of the administrative tasks.

All the activities, except the campus exploration, were performed in a laboratory equipped with 26 computers connected to the Internet. Only the teacher's computer had a Bluetooth interface. Because of such limitation, the teacher uploaded all the log files and used the *cockpit* (see Chapter 4.3.3) to act on behalf of the students when doing the upload.

7.4.4 Evaluation methodology

The enactment of the orchestrated learning flow generated relevant data that allowed analysing the experience from different perspectives. Each source of information contains data related to a different aspect. The combined analysis of all the data sources leads to the obtained results.

The observations taken from the experience were one of the elements to consider in the analysis. During the experience, one external observer (a volunteer different from the tutors and the researchers) annotated those things he considered relevant. Such information was used by the researchers to get a non biased view of the general aspects in the enactment.

The interview with one of the teachers reflects her view of the experience. For administrative reasons, only one of the teachers that participated in the course was interviewed. Having this limitation in mind, the researchers selected the teacher with the less engineering-focused background. The interview was recorded for its later transcription and analysis. The opinion of the rest of the teachers was captured in a questionnaire they filled after the experience.

The students' perception was captured by a questionnaire they filled after finishing the experience. The questionnaire was anonymous and combined open questions with 1 to 5 Likert scaled questions. After the experience, 31 answered questionnaires were available for their analysis.

The above described data were evaluated following the mixed evaluation method described in [11] that combines qualitative and quantitative analysis. This is a case study based method of analysis. That is, the goal is to study how the different factors affected the experience without the aim of generalizing the results. It does not mean that the conclusions cannot be generalized, but such generalization lies on the subjectivity of the researchers. The method proposes to apply *triangulation* to reinforce the findings of the experience. For example, the conclusions extracted from the interview are compared with the questionnaires so that they can be reinforced or invalidated. Thus, the use of different sources of information reduces the inherent subjectivity of the findings.

The answers to numeric questions are represented by their average and standard deviation. The validity of these parameters is given by the performed statistical analysis: the *t-test* was used to determine the 95% interval of confidence for the mean. This type of tests offer more conservative results, but with more confidence on their generalization to a larger population. In the cases where the results of two students' subgroups were compared, the *t-test* was used to determine the statistical significance of the comparative.

A case of study is a complex entity and its analysis might be biased by the different factors that affect the activities and the participants. The analysis of the data includes the identification of these factors, which are presented at the end of the results section. On the researchers' opinion, the influence of these factors has not biased the obtained results but they are presented here to let the reader to make his/her conclusions.

The analysis of the data is try to determine if the orchestration method overcame the limitations identified in the previous experience, explained in Section 7.4.1, and if the requisites imposed in Section 7.4.2. The specific research questions are:

RQ1) Does the orchestration method solve the limitations detected for the CSCBL script (i.e. adaptability, flexibility, integration and scalability)?

RQ2) Does the CSCBL script support teachers' tasks?

RQ3) Does the CSCBL script support students' tasks?

RQ4) Which limitations were detected regarding the orchestration process and its technological support?

7.4.5 Results

The first of the presented research questions regards to what extent was the proposal a solution for the limitations of the previous experience. The analysis is therefore focused on the scalability, adaptability, flexibility and integration.

For the analysis of the scalability, it has to be considered the limitation imposed by the collaborative pattern. That is, the work in groups encouraged by the learning flow best suits for groups of a certain size, and loss its pedagogical meaning for larger groups. As a consequence, the collaborative pattern imposes a limit of 5 members per group, while the number of buildings to study sets the number of groups to 5. Thus, the number of supported participants is 25. To support the participations of more students, more course replicas can be instantiated. Scalability is then related to the replicability of the process. In other words, the scalability of the orchestration method is related to the cost of the replication process. In the experience, four course replicas were held and the instantiation process was performed

during the break between replicas. All the courses started in time, what means that the replication process was not an obstacle for the crowded schedule presented in Table 7.13. The absence of observations or comments shows that neither the teachers nor the students perceived any connexion between course replicas. The interviewed teacher said that: "*The course I taught had no relationship with other courses, or at least this is what I perceived*". The demonstrated replicability of the orchestration method justifies the scalability of the process.

	Question	**Legend**	**Mean**	**Std.**	**Confidence interval for the mean**
01	Does the time between activities breaks the overall pace of the flow?	1 = no, it does not break the pace 2 = yes, it breaks the pace	2.05	1.07	[1, 2.45]
06	Considering the performed task, asses how appropriate your teammates were	1 = Very inappropriate 5 = Very appropriate	4.19	0.75	[3.82, 5]
10.a	Asses the how difficult was to identify the activity you had to do	1 = Very easy 5 = Very difficult	2.35	1.28	[1, 2.74]
10.b	Asses the how difficult was to go to the next activity once you have finished	1 = Very easy 5 = Very difficult	2.32	1.4	[1, 2.67]
10.c	Asses the how difficult was to understand what to do	1 = Very easy 5 = Very difficult	2.29	1.29	[1, 2.68]
12	Asses the perceived integration of the different technologies and activities	1 = bad integration 5 = good integration	3.84	0.68	[3.63, 5]

Table 7.14: Summary table of the numeric answers given to the questionnaires

The group formation process takes place in the course enactment and its analysis lies on the side of the adaptability. Two factors need to be considered at this point: first, the difficulty of storing all the information to the group management tool (the spreadsheet) and, second, the correctness of the decisions taken in such tool.

The teachers, who were supported by the researchers in the group formation process, recognized that they would have not been able to manage the groups without previous training, but they also stated that the process was easy to understand and a training session would have been enough for them to learn the *how to*. For example, one teacher said that "*I guess that if I have had a training course before the actual session, I would have been able to complete the tasks without problem*". The use of a spreadsheet, a tool that the teachers already knew, was useful to hide the complexity of the adaptation scheme and increased the usability of the model. One of the concerns of the researchers was to determine if the time required by the group formation process was excessive for such a classroom activity. Both the teachers and students agreed on saying that the required time did not break the pace

of the activity flow. The former said that "*even a longer time can be tolerated if you have a pre-programmed activity to fill this gap*", while the latter's opinion is reflected in question 01 from Table 7.14.

To measure the correctness of the created groups, the results to question 06 presented in Table 7.14 show that the students thought that the group members were appropriately selected. From their comments in open questions it can be said that they did not realized the actual grouping criteria, but if they should realize it or not is out of the scope of the orchestration method and lies on the pedagogical foundations of the learning flow.

	Question	**Allowed options**	**Answers**
11	Have you need the help of the teacher at any moment	Yes	48.39%
		No	51.61%
		N/A	-
15.a	Which activity did you prefer?	Mobiles	54.58%
		Web	42.42%
		Groups	3%
17	Would you recommend this course to your friends?	Yes	93.56%
		No	3.22%
		N/A	3.22%
18	Would you repeat the course?	Yes	51.61%
		No	48.39%
		N/A	-

Table 7.15: Summary table of the numeric answers given to the questionnaires

Flexibility is a well-known limitation of IMS LD. The provided orchestration method adds two elements intended to increase the level of flexibility offered by the method: the *cockpit* and the use of *spreadsheets* to manage groups. Table 7.16 compiles the unexpected situations that arose during the experience and how they were solved. It can be seen that the most common problem regarded the collaborative nature of the activity flow. The absence of teammates and even entire groups were handled by the *cockpit*, which was used to act on behalf of missing students. The spreadsheet was useful to manually assign the groups in those cases where the actual situation did not fit the ideal case. Despite the unexpected events and according to students' opinion expressed in question 17 and 18 (Table 7.15), the course was successfully enacted and it can be concluded that the flexibility offered by the orchestration method covered the needs of the learning flow.

The experiences taken in prior to the definition of GSI (Chapter 6) revealed that the lack of integration among tools was an obstacle for the successful course enactment. The answers to question 12 (Table 7.14 show that an integrated system was perceived by the students. The teachers suggested that it would have been better if the results of the last questionnaire (QTIMaps based) were stored in the spreadsheet with the rest of the data. That is, without knowing what GSI is and what functionality it provides, they demanded integration in those cases where it has not been provided by the orchestration method. This reveals that GSI offered integration in a transparent way so that the participants did not perceive it, but they missed it when it was not there.

The analysis of the questionnaires shows that the students did not have remarkable problems of following the activity flow. In other words, the orchestration method was able to support students' tasks. Answers from 10.a to 11 reveal that the participants found the system easy to use. Table 7.17 show the comparative on the perceived difficulty of the

issue	**solution**	**tool**
only on computer had Bluetooth interface	the teacher uploaded all logs and updated students properties	cockpit
two students returned the mobile phone after the deadline	they were manually assigned to a group, and their log's data were ignored	spreadsheet
one student dropped the course	the user was ignored in the forthcoming activities	no action needed
only 3 students were in the course	the group formation was manually performed, with the 3 students in the same group	spreadsheet
only one experts' group in the collaborative phase	documents from previous course replicas was used as faked groups in the reflection phase	cockpit

Table 7.16: Summary of unexpected situations and applied solutions.

system among the students who needed teacher's help and those who did not. Despite the former perceive a less usable system, the average of the answers reveals that their perception is positive.

The teachers said that it was easy to determine what to do at each activity. However, they had preferred to prepare these activities in advance (e.g. the day before) and, as stated previously, they demanded a training session to master the system. Usability is more relevant for teachers because, as a teacher said, "*it would have been embarrassing if I had got stuck in one activity without knowing what to do, with my students looking at me*". Apart from these considerations, they found that the system covered their requisites and found it appropriate for the orchestration. It can be said, as the answer to RQ2 and RQ3, that the system was able to support participants' tasks.

	Asked the teacher	**Did not ask the teacher**	**p-value (unpaired samples, one-sided Wilcoxon test)**
10.a	2.87	1.87	0.01258
10.b	2.67	2	0.03662
10.c	2.8	1.81	0.01684

Table 7.17: Comparative among students who asked the teacher and those who did not.

Regarding RQ4, the main limitation found is the abovementioned lack of training of the teaching staff. In this proof of concept, the researchers supported them during the enactment, but training sessions would be required for future course replicas. The lack of integration in the last questionnaire was another flaw detected by the teachers. This fact was known by the researchers, and the teachers' opinion emphasized it. As argued above, such perception of the lack of integration can be considered a flaw of the experience, but a positive fact for the evaluation of GSI, because the demanded functionality is precisely what GSI provides.

Finally, the students proposed some (in their opinion) improvements to the experience, especially for the *Explore the campus* phase. For example, one student said that augmented reality would have been more engaging than NFC interaction. Some students suggested combining the exploration with collaborative activities. One student suggested performing the exploration in groups, while another student said that he would have liked a kind of a *gymkhana* to do the exploration.

Factors that may have biased the data

After the experience, the researchers tried to identify what factors have affected the experience, and if they have biased the results.

The bad weather conditions during the experience were an obstacle for the execution of the exploration with mobiles. This fact discouraged the students and, as a result, the number of participants was lower that what the researchers expected. It could be said that such a low number cannot be used to justify the scalability of the orchestration method. However, the scalability is provided by the high replicability the learning flow. That is, since the creation of a new course replica is almost costless, it is easy to provide support to more students.

Related to the same fact, another factor was the few number of students that took part in the third and fourth replicas. Being only three students, it is possible that they had a different view of the experience. The analysis of the results shows no statistical difference among the answers of those students and the participants of the rest of the course replicas. The only difference found is that the percentage of students who would like to repeat the course was higher in the two last course replicas.

Finally, there was a problem regarding the questionnaires: in the Likert-scaled questions, the positive opinion was associated to the higher values. However, in questions 10.a, 10.b, 10.c and 22, the lower numbers expressed a positive opinion. It is the researchers' opinion that this fact confused the students so, in the referred questions, that they could have marked a high number to express a positive opinion. Actually, these are the questions whose answers have a higher standard deviation, which reinforces the researchers' thoughts. In any case, the opinions collected in the problematic questions are considered positive.

7.4.6 Conclusions

The experience presented in this section is an evaluation of IMS LD and GSI when they are applied to complex collaborative activities that involve different technologies, activities and spatial locations. The method showed a high replicability, which justifies the scalability of the model. That is, the method allow to orchestrate learning flows that otherwise could have not been deployed because of their requirements in terms of administrative workload. The adaptability and flexibility of the model allowed dynamic adjustments in the flow and the ability to manage unexpected situations.

The main lesson learned from the design and enactment of the learning flow is that better results are achieved if the tasks are performed in the tools that were designed for them. That is, in previous experiences, the groups were formed with an *imsldcontent* created for the case and the course showed a poor usability. In the *Meeting the campus* experience, the group management was done in a specialized tool such as Google Spreadsheets, resulting in a more usable and realistic approach.

Generic Service Integration was used to automate the exchange information among the different tools used in the course and to programmatically instantiate the required tools. The integration offered by GSI has allowed the use of third-party tools, increasing the reusability of the model due to the sensitive reduction of the administrative tasks. The analysis of the

experience shows how the combined use of IMS LD and GSI has solved the limitations found in the previous experience.

7.5 Conclusions

Three experiences oriented towards the analysis of the feasibility of GSI in real learning scenarios are presented in this chapter. The implications of the proposed model have been studied: the first of the experiences focused on understanding the practical issues of the model, the second one was oriented towards the support of large-scaled traditional learning models and the last experience analysed the appropriateness of the proposed system in innovative scenarios.

The enactment of the experiences provided significant data to discuss, from the practical point of view, the requisites of the model stated in 5.2.1 and theoretically analysed in 5.5. They are *Pedagogical neutrality*, *Reusability*, *Self-containment*, *Adaptability* and *Collaborative capabilities*.

Pedagogical neutrality. Different learning methods were applied in each of the presented experiences. That is, different pedagogical models have demonstrated their applicability in GSI courses. Furthermore, the *GSpread* service adapter has been used for different purposes in the first and the last experiences, showing the versatility of the integrated tools.

Reusability. The creation of an infrastructure to share and reuse learning courses was out of the scope of this dissertation, so the reusability of UoLs has not been studied in such context. However, the second experience revealed the feasibility of reusing IMS LD templates to speed up the creation process. The use of *GSpread* for different purposes also showed the reusability of service adapters with different pedagogical approaches.

Self-containment. This characteristic is related with the replicability of courses, which has been shown to be one of the strengths of the GSI model. Apart from justifying self-containment, the demonstrated replicability asserts the scalability of courses in large-scaled scenarios.

Adaptability. The adaptive capabilities offered by IMS LD are upgraded by GSI with the support of external sources of information. This assertion was empirically demonstrated with the presented experiences. For example, in programming systems the students' schedule depended on the result of the tests performed in the third party tool.

Collaborative capabilities In the programming course experience, the students were working in pairs. A more sophisticated approach drove the *Meeting the campus* scenario, where the students worked in dynamically-created groups. In both cases, the collaborative nature of the pedagogical model was enacted with the support of the GSI technology.

Chapter 8

Conclusions

> Doubt comes in at the window when inquiry is denied at the door.
>
> Benjamin Jowett

This dissertation presented a research work that explores the limits of the IMS LD specification and proposes empirically validated solutions for the encountered limitations. This chapter concludes the document with a summary of the most relevant contributions of the presented work. Also, the text presents interesting research questions regarding the proposed model that have not been addressed in this dissertation but could be adopted as guidelines for future research.

8.1 Contributions

Practical experiences with IMS LD

The first phase of the adopted research methodology was the information gathering. The analysis of the state of the art revealed a relative high number of theoretical studies of the IMS LD expressiveness. However, the literature regarding practical experiences with the specification did not allow the complete characterization of the course life-cycle. This dissertation characterized the course life-cycle with the deployment of field experiences, which also assessed the expressiveness of IMS LD in real courses.

The first of these experiences focused on the analysis of IMS LD expressiveness with real-world examples. Most of the existing courses found in the literature were synthetic learning flows created as the proof of concept of one of the IMS LD capabilities. The experience considered an already existing course, being taught in an engineering course, and translated it to the IMS LD terminology. Then, the deployment and enactment of the resulting UoL was simulated so that the characteristics of the enactment process could be theoretically identified. A lesson learned from the experience was the complete knowledge of the specification that is required for the course authoring. Also, the experience revealed that the expressiveness of IMS LD goes beyond the theatrical metaphor [82].

The second experience included the enactment phase of the UoL. In such experience, students from three geographically distributed higher education institutions participated in a course where the collaboration among participants played a relevant role in the learning flow. Students' interactions were modelled and delivered using IMS LD. The goal of the study

was to identify the feasibility of the model to support collaborative activities in distance scenarios. The results showed that Web based tools are required to support collaborative activities, but the lack of support of such tools in the IMS LD framework reduced the replicability of the created course. The experience was held in two editions: the first one, enacted by volunteer PhD. students, used roles to model the groups that participated in colaborative activities. The main problem encountered during the experience was the need to manually administer the third-party tools, which resulted in a high and error prone administrative workload [207]. The second experience was enacted with real students. It revealed the difficulty of course replication when the activity sequence incorporates the use of third-party tools [115, 208]. Besides, the experience presented the flow of artifacts in complex collaborative situations as one of the limitations of IMS LD [116].

Finally, the third experience consisted in the design and implementation of mashups support in a IMS LD player. In such work, IMS LD was used for the high level orchestration of learning activities, while a more specific tool provided orchestration at the level of fine-grained tasks. The proof of concept implementation showed how IMS LD is oriented towards the orchestration of learning activities, while has problems if it is used to model the specific details of a single activity. The use of case-specific tools for the modelling and enactment of the particular details of an activity improved the expressiveness of the IMS LD framework [215].

Improvements of IMS LD support in .LRN

The expertise gained from the implementation of GRAIL [10] and the deployment of field experiences with IMS LD allowed to identify limitations of the model, whose solution was proposed and implemented in the context of this dissertation.

Chapter 4 presented the technical description of GRAIL, the IMS LD player that was used in all the experiences presented in this dissertation. This chapter also presented the functionality included in the administrator's interface of GRAIL aimed at increasing the flexibility of the enactment phase. Enactment flexibility in IMS LD courses is defined as the capability to react to unexpected situations without losing the intrinsic constraints of the learning script. The provided solution enables the modification of the course and distinguishes among two different situations [180]:

- Learning flow modification. The administrators' interface supports the modification of activities' completion conditions and the inclusion/removal of learning activities in the activity sequence. The inclusion and edition of environments' resources in also supported.
- Content modification. The use of wiki technology to manage HTML files in the IMS LD player allows for the content modification right from the user's interface. A permissions system managed by the course administrator dictates who is able to perform such modifications. Such functionality enables new scenarios such as the collaborative authoring of UoLs.

The improvements in GRAIL were documented and constrasted in several international forums [146, 250, 118, 181]. The participation in dissemination events and/or research workshops enriched the author's knowledge and perspective of the specification, and influenced the decisions taken during the design of the GSI framework [216, 84, 189].

Definition of the *Generic Service Integration* Framework

The definition and implementation of the proposal were, respectively, the second and third phases of the adopted methodology. The definition of *Generic Service Integration* covers

the complete course life-cycle:

1. The IMS LD vocabulary is augmented with elements aimed at the inclusion of third-party tools in the learning flow. The provided vocabulary defines *what* type of tool is going to be needed, *who* (which role) will use it and *how* it will be used.
2. In the deployment phase, the abstract service definition included in the manifest is used to select the third-party tool that best matches the author requirements. The list of the available tools is composed by those tools whose support is provided by the corresponding service adapter installed in the platform.
3. In GSI, the tool is not an activity, it is the support required to perform the activities. A link to the third-party tool is placed in the corresponding environment so that the users can access the tool during the course enactment. Besides, the IMS LD events may cause the player to request information from the third-party tool and use this information to adapt the course flow, or whatever use given to the course properties.

The integration of a specific tool is provided by the development of a service adapter that mediates the communication between GSI and the external tool [190]. The proposed architecture promotes the quick development of new service adapters.

There exist some proposals in the state of the art aimed at the integration of services in IMS LD courses (*Gridcole*, *CopperCore Service Integration Layer*, *Wookie* or *IMS Tools Interoperability*). *Generic Service Integration* was defined as a response to the need of a framework that offer the following characteristics:

- Is not restricted to tools that use a particular comunication protocol. *GSI* service adapters allow the inclusion of third-party tools that use proprietary or open APIs.
- Provides UoLs with *self-containment*, *interoperability* and *replicability*. That is, a course that uses third-party tools can be enacted in different platforms without significant changes in the learning flow.
- Enables, during the enactment, the bidirectional exchange of information between the IMS LD workflow engine and the third-party tool.
- Considers the implications of service integration in the complete course life-cycle: authoring, deployment and enactment.

Analysis of identity and authentication issues

A detailed analysis revealed that the authentication needs hinder the practical deployment of third-party tools integration. The problem can be summarised as follows: if the IMS LD player is going to retrieve information stored in a third-party tool that belongs to the user, it is required for such user to explicitly grant access to the IMS LD player in the external tool. Such administrative task should not interrupt the course enactment, so that it should be performed during deployment. However, the users may not be available at the deployment phase so they would not be able to complete the task.

The GSI model proposes the inclusion of a new phase in the course life-cycle, called the pre-enactment and included in the course by means of the *act zero* [193]. The *act zero* is an activity automatically included in the sequence, and whose completion is required in order to start performing the actual learning activities. The content of the act zero, as well as the roles who have play it, depends on the tool selected during the deployment phase. More precisely, it is the service adapter who includes such act and these activities. Typically, the act zero will inform the user of the use of external services and will offer a method to manually grant access to the external service.

The implementation of the framework

Chapter 5 provided technological details of the GSI support developed for the GRAIL player. GSI complements the already existing player so that it allows the enactment of of GSI-shaped UoLs without interfeering the deployment and enactment of the rest of the UoLs. The extension, as well as the whole player, was published under open source license.

Two service adapters were developed and installed in the player. That is, the player provides support for the integration of two different services. First, the *GSpread* adapter integrates Google Spreadsheets functionality: the answers of a questionnaire posed to the students are stored in a spreadsheet, which is accessed via Web and hosted by Google. After its manipulation in the spreadsheet, IMS LD retrieves the data and sets the value of the corresponding properties [191]. The second developed adapter integrates software testing capabilities in IMS LD courses: after developing a piece of code proposed by the teacher, the students upload it to a server that tests its functionality and returns true or false, depending on the result of the test. Such return value can be captured by the IMS LD player so that the course flow can react to the students' results.

Experimental work with GSI

The last phase of the research methodology consisted in the experimental validation of the proposed solution. In this dissertation, the validation was performed by means of three cases of study aimed at determining to what extent the proposed framework satisfies the requirements stated in Section 5.2.1 and theoretically discussed in Section 5.5.

The first experience focused on two aspects: first, it analysed the feasibility of the proposed framework; second, offered a perspective of the instructors understanding of the model. A workshop with volunteers was conducted so that the participants were trained in the use of IMS LD and GSI with the *GSpread* service adapter. The deployment process revealed a high replicability of GSI-shaped courses [202]. This replicability was an important factor for the success of the subsequent experiences. One lesson learned is that the instructors found no special difficulty on the understanding of the GSI framework. Also, the analysis of the results confirmed that the use of a specialised third-party tool (i.e. a spreadsheet) to manage the data increases the usability of the framework.

The second experience, deployed in a large scaled scenario, was devoted to the analysis of the performance of the framework and its expressiveness when applied to traditional learning methods. A learning flow based on the Project Based Learning methodology was followed by the students of a programming course held in the second term of the University Carlos III of Madrid. The *delivery-platform* service adapter was used so that the students were able to access to the next activity in a project only when they passed the test of the activity under development. A total of 425 students and 8 teachers participated in the course. The analysis of the case of study showed that the orchestration method promoted the students' continuous work without increasing the workload of the teaching staff. That is, the combined use of IMS LD and GSI was suitable for the management of large-scaled scenarios [183]. According to the the questionnaire answered by the students, two out of three of them would like to use the same system in the future.

Finally, the deployment of the *Meeting the campus* case of study analysed if the orchestration method consisting on the use of IMS LD and GSI is able to reduce the administrative workload inherent to complex sequences of activities. This experience was constructed upon a previous existing case of study that mixed the use of mobile phones, collaborative learning activities, different spatial locations, and informal activities. The combined use of such ingredients offered encouraging results in terms of students' motivation and learning benefits, but imposed a too high administrative workload that hindered the replication of the course.

The semi-automation of the administrative task was conducted by the use of the *GSpread* service adapter, so that the management of students data was performed in a spreadsheet while the IMS LD script automated the group management and the flow of artifacts [182]. An analysis of the data offered by the case of study showed that the automatically orchestrated version of the course was pedagogically equivalent to the previous one, with the advantage that imposed less administrative workload. The course flow was enacted four times with different actors, demonstrating the replicability of GSI-shaped courses and the scalability of the framework proposed in this dissertation.

8.2 Future research directions

The work presented in this dissertation explores the use of third-party services in the context of IMS LD scripted courses. The dissertation studies the problem from a holistic point of view, trying to identify the needs of the model in the course life-cycle. This work has identified open ended research problems that are presented as follows:

- To use patterns to help practitioners in the authoring process. GSI defines a vocabulary that enables the inclusion of third-party tools in the description of the learning flow given in the manifest file. However, the only available authoring method is the inclusion of the XML elements with a text editor. It is unrealistic to think that this method could be adopted by practitioners and the experience with IMS LD show the relevance of a user friendly authoring tool. Such tool should provide an answer for the following questions: which is the more adequate tool description that can be included in a course? how do course authors reach the more adequate description?

  Authoring collaborative activities with *Collaborative Learning Flow Patterns* (CLFPs) revealed the usefulness of patterns at course authoring. The same idea could be considered for the definition of third-party tools. This functionality would prevent course authors to include incomplete descriptions of the required services and would also advertise the existing services to the public.

- Automatic selection of services at deployment. According to GSI, the generic description of the third-party tool that is included in the manifest file is used during the deployment to choose the best matching available tool. In the experiences presented in this dissertation, the tool selection has been manually done, being the course administrator the responsible of the task. This method suited for the presented experiences, because the course administrator was a member of the research team, so that he was an expert in the field.

  However, course administrators do not always have the required pedagogical skills to complete the task. For example, they might be mere technical administrators of the platform with no pedagogical background and they might not be the most adequate actor to select the third-party tool. The vocabulary offered by GSI includes a human readable description of the expected tool. Thus, semantic techniques could be used in order to find the most adequate service and properly configure the course.

- A distribution method for available adapters. According to the proposed architecture, GSI is a pluggable framework where the support of a specific tool is provided by the installation of the proper service adapter. This dissertation discusses the feasibility of the idea and provides an implementation for the GRAIL player. In order not to limit the list of available services to the already installed service adapters, an easy-to-find-and-install approach is worth to be explored.

The idea has similarities with software repositories such as *Android Market* or *iTunes* for mobile phones, where the end user is provided with an easy interface to find new applications and install them. Further research on this idea should consider interoperability issues and how the distribution methods can be related with automatic selection of services.

- Support during enactment for flexibility of bidirectional information exchange. GSI enables IMS LD to request information from the third-party service. According to the proposed model, the structure of the gathered information is offered to the GSI layer as a set of arrays. The type of information stored at each array's position is explained in the service adapter's documentation, as explained in Chapter 5.2.2. When a course author needs to refer to the third-party tool's information, he/she states the position where, according to the documentation, the desired information is stored.

  This method lacks of flexibility and reusability of tool definitions. It would be desirable to provide players with a method to edit the array's position from where the information will be gathered. In summary, the research would face the following question: how to provide a flexible and reusable method that allows generic definitions to retrieve specific information stored in third-party platforms?

- Extrapolation of third-party tool integration to different contexts. The framework proposed in this dissertation was conceived for its use in IMS LD scripted courses. The course life-cycle imposed by the specification has conditioned the shape and architecture of the proposal. The use of IMS LD was intended to provide automatic orchestration methods for the activities supported by third-party tools.

  The relevance of tools' integration goes beyond the IMS LD specification and it is therefore worth to explore the application of the model in different context. One example is the use of different scripting languages for learning courses. Another approach is analysing the impact of tools' integration in the existing *Learning Management Systems*, where the course life-cycle is radically different to the IMS LD one.

Bibliography

[1] Jean Piaget. *The Moral Judgment of the Child*. Routledge & Kegan Paul, London, 1932.

[2] L. S. Vygotsky. *Mind in Society: The Development of Higher Psychological Processes*. Harvard University Press, 1978.

[3] David Wiley. Connecting learning objects to instructional design theory: A definition, a metaphor, and a taxonomy. *The instructional use of learning objects*, 2830(435):1–35, 2000.

[4] Xavier Ochoa. *Learnometrics: Metrics for Learning Objects*. PhD thesis, September 2008.

[5] David Wiley. RIP-ping on Learning Objects. Iterating toward openness, 2006. http://opencontent.org/blog/archives/230, last visited 2011-01-29.

[6] IMS Global Learning Consortium. IMS Learning Design Specification, February 2003. Last Visited, January 2011 at http://www.imsglobal.org/learningdesign/.

[7] Paul Anderson. What is Web 2.0?: ideas, technologies and implications for education. Technical report, JISC Technology and Standards Watch, 2007.

[8] Herwig Rollett, Mathias Lux, Markus Strohmaier, Gisela Dosinger, and Klaus Tochtermann. The Web 2.0 way of learning with technologies. *International Journal of Learning Technology*, 3(1):87 – 107, 2007.

[9] W. Richards Adrion. Research Methodology in Software Engineering, in Future directions in software engineering, summary of the Dagstuhl Workshop. *SIGSoft Software Engineering Notes*, 18(1):36–37, 1993.

[10] Jose Pablo Escobedo Del Cid, Luis de-la-Fuente-Valentín, Sergio Gutiérrez, Abelardo Pardo, and Carlos Delgado Kloos. Implementation of a learning design run-time environment for the .LRN Learning Management System. *Journal of Interactive Media in Education Special Issue: Adaptation and IMS Learning Design*, September 2007.

[11] Alejandra Martínez-Monés, Yannis Dimitriadis, Bartolomé Rubia, Eduardo Gómez, and Pablo De La Fuente. Combining qualitative evaluation and social network analysis for the study of classroom social interactions. *Computers & Education*, 41(4):353–368, 2003.

[12] Ryan Watkins and Doug Leigh, editors. *Handbook of Improving Performance in the Workplace: Selecting and Implementing Performance Interventions*. John Wiley & Sons, Inc., Hoboken, NJ, USA, November 2009.

[13] Scott Wilson, Oleg Liber, Mark Johnson, Phil Beauvoir, Paul Sharples, and Colin Milligan. Personal Learning Environments: Challenging the dominant design of educational systems. *Journal of e-Learning and Kowledge Society*, 3(2):67–76, October 2006.

[14] Mark van Harmelen. Personal Learning Environments. In *Sixth IEEE International Conference on Advanced Learning Technologies*, pages 815–816. IEEE, 2006.

[15] France Henri and Bernadette Charlier. Personal learning environment: A concept, an application, or a self-designed instrument? In *Information Technology Based Higher Education and Training (ITHET), 2010 9th International Conference on*, pages 44 –51, May 2010.

[16] Edutech Wiki. Learning Management Systems, 2006. Last visited, January 2011 at http://edutechwiki.unige.ch/en/Learning_management_system.

[17] Michael Machado and Eric Tao. *Blackboard vs. moodle: Comparing user experience of learning management systems.* IEEE, October 2007.

[18] Elsebeth Korsgaard Sorensen and Daithi O. Murchu, editors. *Enhancing Learning Through Technology.* Information Science Publishing, 2006.

[19] Govindasamy Thavamalar. Successful implementation of e-Learning: Pedagogical considerations. *The Internet and Higher Education*, 4(3-4):287–299, 2001.

[20] Mildred Roqueta. Learning management systems: A focus on the learner. *Distance Learning*, 5(4):59–66, 2008.

[21] Benjamin S. Bloom. *Taxonomy of Educational Objectives, Handbook 1: Cognitive Domain.* Addison Wesley Publishing Company, 1956.

[22] Andrew Churches. Bloom's Digital Taxonomy, 2008. Last visited, January 2011 at http://edorigami.wikispaces.com/Bloom

[23] Stephen Downes. E-learning 2.0. *eLearn magazine*, 2005(10), 2005.

[24] Carsten Ullrich, Kerstin Borau, Heng Luo, Xiaohong Tan, Liping Shen, and Ruimin Shen. Why web 2.0 is good for learning and for research: principles and prototypes. *International World Wide Web Conference*, pages 705–714, 2008.

[25] Eleni Kaldoudi, Panagiotis Bamidis, Miltiadis Papaioakeim, and Vassilis Vargemezis. *Problem-Based Learning via Web 2.0 Technologies.* IEEE, June 2008.

[26] Timothy J. Ellis and Maxine S. Cohen. Forums and wikis and blogs, oh my: Building a foundation for social computing in education. In *IEEE Frontiers in Education Conference*, pages 1–2. IEEE, October 2009.

[27] Nauman Saeed, Yun Yang, and Suku Sinnappan. *Effects of Cognitive Style on User Acceptance of Blogs and Podcasts.* IEEE, July 2009.

[28] Learning for tomorrow's world. First results from PISA 2003. Technical report, Programme for International Student Assessment (PISA), 2004.

[29] Ralf Klamma, Marc Spaniol, Yiwei Cao, and Matthias Jarke. Pattern-Based Cross Media Social Network Analysis for Technology Enhanced Learning in Europe. In *Innovative Approaches for Learning and Knowledge Sharing*, volume 4227 of *Lecture Notes in Computer Science*, pages 242–256. Springer Berlin / Heidelberg, 2006.

[30] Ilaria Liccardi, Asma Ounnas, Reena Pau, Elizabeth Massey, Päivi Kinnunen, Sarah Lewthwaite, Marie-Anne Midy, and Chandan Sarkar. The role of social networks in students' learning experiences. *ACM SIGCSE Bulletin*, 39(4):224–237, 2007.

[31] Christian Dalsgaard. Social software: e-learning beyond learning management systems. *European Journal of Open, Distance and E-Learning*, 2006(2), 2006.

[32] Mohamed Amine Chatti, Matthias Jarke, and Dirk Frosch-Wilke. The future of e-learning: a shift to knowledge networking and social software. *International Journal of Knowledge and Learning 2007*, 3(4/5):404–420, 2007.

[33] Mark Maybury, Ray D'Amore, and David House. Expert Finding for Collaborative Virtual Environments. *Communications of the ACM*, 44(12):55, 2001.

[34] Jeremy Goecks and Dan Cosley. NuggetMine: intelligent groupware for opportunistically sharing information nuggets. *International Conference on Intelligent User Interfaces*, page 87, 2002.

[35] Ralf Klamma, Mohamed Amine Chatti, Erik Duval, Hans Hummel, Ebba Thora, Milos Kravcik, Effie Law, Ambjörn Naeve, and Peter Scott. Social Software for Life-long Learning. *Educational Technology and Society*, 10(3):72–83, 2007.

[36] Linden Research, Inc. Second Life, 1999. Last visited, November 2010 at http://www.secondlife.com.

[37] Steven Warburton. Second Life in higher education: Assessing the potential for and the barriers to deploying virtual worlds in learning and teaching. *British Journal of Educational Technology*, 40(3):414–426, May 2009.

[38] María Blanca Ibanez, José Jesús García, Sergio Galán, David Maroto, Diego Morillo, and Carlos Delgado Kloos. Multi-User 3D Virtual Environment for Spanish Learning: A Wonderland Experience. *IEEE International Conference on Advanced Learning Technologies*, pages 455–457, 2010.

[39] Peggy A. Ertmer and Timothy J. Newby. Behaviorism, Cognitivism, Constructivism: Comparing Critical Features from an Instructional Design Perspective. *Performance Improvement Quarterly*, 6(4):50–72, October 1993.

[40] Carl Dreher, Torsten Reiners, Naomi Dreher, and Heinz Dreher. *3D virtual worlds as collaborative communities enriching human endeavours: Innovative applications in e-Learning*. IEEE, June 2009.

[41] Open Wonderland, 2010. Last visited, January 2011 at http://openwonderland.org.

[42] Bill Olivier. Lifelong Learning: The Need for Portable Personal Learning Environments and Supporting Interoperability Standards. 2001.

[43] Ge Jian Ding. *A preliminary study of personal learning environment based on Ubiquitous Computing Model*. IEEE, July 2010.

[44] Graham Atwell. Personal Learning Environments-the future of eLearning? *E-Learning Papers*, 2007.

[45] Edutech Wiki. Personal Learning Environments at Edutech Wiki, 2007. Last visited, November 2010 at http://edutechwiki.unige.ch/en/Personal_learning_environment.

[46] Oleg Liber. Colloquia – a conversation manager. *Campus-Wide Information Systems*, 17(2):56–62, January 2000.

[47] Teemu Leinonen, Otso Virtanen, Kai Hakkarainen, and Giedre Kligyte. Collaborative Discovering of Key Ideas in Knowledge Building. In *Proceedings of Computer Support for Collaborative Learning Conference*, 2002.

[48] Feng Wang, Xiayuan Li, Chengling Zhao, and Chunyan Xu. Construct Personal Learning Environment Based on Web2.0. In *International Conference on Management and Service Science*, pages 1–4. IEEE, September 2009.

[49] Lino Oliveira and Fernando Moreira. Integration of Web 2.0 applications and content management systems on personal learning, environments. *Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–5, 2010.

[50] Vlad Posea and Stefan Trausan-Matu. *Bringing the Social Semantic Web to the Personal Learning Environment*. IEEE, July 2010.

[51] Webmashups, the open directory for Mashups and Web 2.0 APIs, 2006. Last visited, January 2011 at http://www.webmashup.com.

[52] Charles Severance, Joseph Hardin, and Anthony Whyte. The coming functionality mash-up in Personal Learning Environments. *Interactive Learning Environments*, 16(1):47–62, April 2008.

[53] Special Issue: ICL2009 – MashUps for Learning. International Journal of Emerging Technologies in Learning, 2009. http://online-journals.org/i-jet/issue/view/78.

[54] 3rd European Conference On Technology Enhanced Learning. Workshop on Mash-Up Personal Learning Environments (MUPPLE'08), 2008.

[55] Maan M. Shaker, AbdulSattar M. Khidhir, and Younis A. Al-Rizzo. Design and implementation of a virtual university prototype technology requirements and evaluation. In *Proceedings of 4th International Multiconference on Computer Science and Information Technology*, pages 560–568, 2006.

[56] Carol Fallon and Sharon Brown. *E-Learning Standards: A Guide to Purchasing, Developing, and Deploying Standards-Conformant E-Learning*. CRC Press, 2002.

[57] IMS Global Learning Consortium, 1997. Last visited, January 2011 at http://www.imsglobal.org.

[58] Aviation Industry CBT Committee. Last visited, November 2010 at http://aiccc.org/.

[59] Advanced Distributed Learning initiative. Last visited, November 2010 at http://www.adlnet.gov.

[60] The Advanced Distributed Learning initiative. Sharable Content Object Reference Model(SCORM), 2004.

[61] Open Knowledge Initiative, 2001. Last visited, November 2010 at http://www.okiproject.org/.

[62] CEN Workshop on Learning Technologies, 1999. Last visited, March 2011 at http://www.cen-ltso.net/Main.aspx?AspxAutoDetectCookieSupport=1.

[63] John J. Sparkes. Quality in engineering education. *International Journal of Continuing Engineering Education and Life-Long Learning*, 1(1):18–32, 1990.

[64] Phyllis C. Blumenfeld, Elliot Soloway, Ronald W. Marx, Joseph S. Krajcik, Mark Guzdial, and Annemarie Palincsar. Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational Psychologist*, 26(3):369–398, 1991.

[65] Marjahan Begum. Book Review: Research on PBL Practice in Engineering Education. *Interdisciplinary Journal of Problem-based Learning*, 4(2), 2010.

[66] David W. Johnson, Roger T. Johnson, and Edythe Johnson Holubec. *Cooperation in the classroom*. Interaction Book Co, 1988.

[67] Sara McNeil. What is Instructional Design? Last visited, 2007-10-19 from http://www.coe.uh.edu/courses/cuin6373/whatisid.html.

[68] Leonard C. Silvern. *Designing Instructional Systems*. Education and Training Consultants, Los Angeles, 1964.

[69] Ludwig Von Bertalanffy. *General System Theory: Foundations, Development, Applications*. George Braziller, New York, 1976.

[70] Robert Reiser and John V. Dempsey. *Trends and Issues in Instructional Design and Technology*. Prentice Hall, 2001.

[71] Michael Molenda. In search of the elusive ADDIE model. *Performance improvement*, 42(5):34–37, 2003.

[72] D. Barnett, A. Bauer, S. Bell, N. Elliott, H. Haski, E. Barkley, D. Baker, and K. Mackiewicz. Preschool intervention scripts: lessons from 20 years of research and practice. *The Journal of Speech-Language Pathology and Applied Behavior Analysis*, 2(2):158–181, 2007.

[73] Pierre Dillenbourg, Patrick Jermann, Armin Weinberger, K Stegman, and Frank Fischer. A framework for integrated learning scripts. Technical report, Kaleidoscope, European Network of Excellence on learning technologies, 2004.

[74] Pierre Dillenbourg and P. Tchounikine. Flexibility in macro-scripts for computer-supported collaborative learning. *Computer Assisted Learning*, 23(1):1–13, February 2007.

[75] Pierre Dillenbourg. Over-scripting CSCL: The risks of blending collaborative learning with instructional design. In *Three worlds of CSCL. Can we support CSCL?*, pages 61–91, 2002.

[76] Pierre Laforcade. A Domain-Specific Modeling approach for supporting the specification of Visual Instructional Design Languages and the building of dedicated editors. *Journal of Visual Languages & Computing*, September 2010.

[77] Rob Koper and Jocelyn Manderveld. Educational modelling language: modelling reusable, interoperable, rich and personalised units of learning. *British Journal of Educational Technology*, 35(5):537–551, 2004.

[78] IMS Global Learning Consortium. IMS Content Packaging Specification, 2001. Last Visited, November 2010 at http://www.imsglobal.org/content/packaging/.

[79] IMS Global Learning Consortium. IMS Learning Resource Meta-data Specification, 2001. Last Visited, November 2010 at http://www.imsglobal.org/metadata/.

[80] Rob Koper and Bill Olivier. Representing the learning design of units of learning. *Educational Technology and Society*, 7(3):97–111, 2004.

[81] IMS Global Learning Consortium. IMS Question and Test Interoperability, February 2003. Last Visited, November 2010 at http://www.imsglobal.org/question.

[82] Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Using learning design to deploy and administer engineering courses. In *Frontiers in Education 2007*, pages S3D–7–S3D–12, Milwaukee, USA, 2007.

[83] Daniel Burgos Solans. *Extensión de la especificación IMS Learning Design desde la adaptación e integración de unidades de aprendizaje.* PhD thesis, University Carlos III of Madrid, April 2008.

[84] Susanne Neumann, Michael Klebl, Dai Griffiths, Luis de-la-Fuente-Valentín, Davinia Hernández-Leo, Hans Hummel, Francis Brouns, Michael Derntl, and Petra Oberhuemer. Report of the Results of an IMS LEarning Design Expert Workshop. *International Journal of Emerging Technologies*, 5(1):58–72, November 2010.

[85] Wim Van der Vegt and Rob Koper. CooperAuthor v1.5, 2006. Last visited, November 2010 at http://dspace.ou.nl/handle/1820/571.

[86] The RELOAD Project, March 2007. Last visited, November 2010 at http://www.reload.ac.uk.

[87] David Griffiths, Phillip Beauvoir, Oleg Liber, and Mark Barrett-Baxendale. From Reload to ReCourse: learning from IMS Learning Design implementations. *Distance Education*, 30(2):201–222, August 2009.

[88] Susanne Neumann and Petra Oberhuemer. Supporting Instructors in Creating Standard Conformant Learning Designs: the Graphical Learning Modeler. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications 2008*, volume 2008, pages 3510 – 3519, 2008.

[89] Yongwu Miao. CoSMoS: Facilitating Learning Designers to Author Units of Learning Using IMS LD. In *Proceeding of the 2005 conference on Towards Sustainable and Scalable Educational Innovations Informed by the Learning Sciences*, pages 275–282. IOS Press, 2005.

[90] Pythagoras Karampiperis and Demetrios Sampson. A Flexible Authoring Tool Supporting Learning Activities. In *Proceedings of the IADIS International Conference on Cognition and Exploratory Learning in Digital Age*, pages 51–58, 2004.

[91] Gilbert Paquette, Michel Léonard, and Karin Lundgren-Cayrol. The MOT+ Visual Language for Knowledge-Based Instructional Design. In *Handbook of Visual Languages for Instructional Design: Theories and Practices. Hershey, PA: Information Science Reference*, pages 133–154, 2008.

[92] James Dalziel. Implementing learning design: The learning activity management system (LAMS). In *Interact, Integrate, Impact: Proceedings of the 20th Annual Conference of the Australasian Society for Computers in Learning in Tertiary Education. Adelaide*, pages 7–10, 2003.

[93] Davinia Hernández-Leo, Eloy Villasclaras-Fernández, Iván M. Jorrín-Abellán, Juan I. Asensio-Pérez, Yannis Dimitriadis, I. Ruiz-Requies, and B. Rubia-Avi. COLLAGE, A collaborative Learning Design editor based on patterns. *Technology & Society*, 9(1):58–71, January 2006.

[94] Patricia López Cuevas, Pedro J Muñoz Merino, Carmen Fernández-Panadero, and Carlos Delgado Kloos. CourseEditor : A Course Planning Tool Compatible With IMS-LD. *Computer Applications in Engineering Education*, (20486), 2010.

[95] Open Universiteit Nederland. CopperCore The IMS Learning Design engine, 2004. Last visited, November 2010 at http://coppercore.sourceforge.net/.

[96] Hubert Vogten, Colin Tattersall, Rob Koper, Peter van Rosmalen, Francis Brouns, Peter Sloep, and Harrie Martens. Designing a Learning Design Engine as a Collection of Finite State Machines. *International Journal on E-Learning*, 5, 2004.

[97] Martin Weller. The SLeD project: Investigating Learning Design and Services. JISC E-learning Focus, 2006. http://www.elearning.ac.uk/features/sledproject.

[98] Miguel L. Bote-Lorenzo, Eduardo Gómez-Sánchez, Guillermo Vega-Gorgojo, Yannis A. Dimitriadis, Juan I. Asensio-Pérez, and Iván M. Jorrín-Abellán. Gridcole: A tailorable grid service based system that supports scripted collaborative learning. *Computers & Education*, 51(1):155–172, August 2008.

[99] E-Lane Project, November 2003. http://www.e-lane.org; Last visited, January 2008 at http://web.archive.org/web/20080113190657/http://e-lane.org/.

[100] The .LRN Consortium. .LRN: Learn, Research, Network. Last visited, January 2011 at http://dotlrn.org.

[101] The Open Architecture Community System, 1999. Last visited, January 2011 at http://openacs.org/.

[102] Juan Vidal, Manuel Lama, Eduardo Sánchez, and Alberto Bugarín. Application of Petri Nets on the Execution of IMS Learning Design Documents. In *Times of convergence. Technologies across learning contexts*, pages 461–466, 2008.

[103] Kim Hagen, Diana Hibbert, and Kinshuk. Developing a Learning Management System Based on the IMS Learning Design Specification. In *IEEE International Conference on Advanced Learning Technologies*, pages 420–424. IEEE Computer Society, July 2006.

[104] Amir Benmimoun and Philippe Trigano. netUniversity: An Interoperable LMS/L-CMS for Adaptive and Collaborative Learning. In *2008 Eighth IEEE International Conference on Advanced Learning Technologies*, pages 666–668. IEEE, July 2008.

[105] Meng-Che Chen, Chien-Tsun Chen, Yn Chin Chen, and Chin-Yun Hsieh. On the development and implementation of a sequencing engine for IMS learning design specification. In *International Conference in Advanced Learning Technologies*, pages 636–640, Kaohsiung, Taiwan, 2005.

[106] Beatriz Fernandez-Gallego, Manuel Lama, Juan Carlos Vidal, Eduardo Sanchez, and Alberto Bugarin. OPENET4VE: A Platform for the Execution of IMS LD Units of Learning in Virtual Environments. In *IEEE International Conference on Advanced Learning Technologies*, pages 472–474, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[107] Imran A. Zualkernan, Sina Nikkhah, and Mohammad Al-Sabah. A Lightweight Distributed Implementation of IMS LD on Google's Android Platform. In *IEEE International Conference on Advanced Learning Technologies*, pages 59–63, July 2009.

[108] Dai Griffiths and Josep Blat. The role of teachers in editing and authoring units of learning using IMS Learning Design. *Journal of Advanced Technology for Learning*, 2(3), 2005.

[109] Daniel Burgos, Colin Tattersall, Martin Dougiamas, Hubert Vogten, and Rob Koper. A first step mapping IMS learning design and moodle. *Journal of Universal Computer Science*, 13(7):924–931, 2007.

[110] Manuel Caeiro-Rodríguez, Luis Anido-Rifón, and Martín Llamas-Nistal. A critical analysis of IMS Learning Design. In *Proceedings of the Computer Supported Collaborative Learning Conference*, pages 363–367, 2003.

[111] Yu Dan and Chen XinMeng. Creating computer supported collaborative learning activities with IMS LD. In *Proceedings of the 12th international conference on Human-computer interaction: applications and services*, pages 391–400, 2007.

[112] Aiman Turani and Rafael Calvo. Sharing Synchronous Collaborative Learning Structures using IMS Learning Design. In *2006 7th International Conference on Information Technology Based Higher Education and Training*, pages 25–34. IEEE, July 2006.

[113] Ricardo Valdivia, Miguel Nussbaum, and Sergio F. Ochoa. Modeling a Collaborative Answer Negotiation Activity Using IMS-Based Learning Design. *IEEE Transactions on Education*, 52(3):375–384, August 2009.

[114] Rocío Garcia-Robles, Fernando Diaz-del Rio, Saturnino Vicente-Diaz, and Alejandro Linares-Barranco. An eLearning Standard Approach for Supporting PBL in Computer Engineering. *IEEE Transactions on Education*, 52(3):328–339, 2009.

[115] Luis de-la-Fuente-Valentín, Abelardo Pardo, Juan I. Asensio Pérez, Yannis Dimitriadis, and Carlos Delgado Kloos. Collaborative Learning Models on Distance Scenarios with Learning Design: a Case Study. In *International Conference on Advanced Learning Technologies*, pages 278–282, Santander, Spain, 2008.

[116] Luis Palomino-Ramírez, Miguel Bote-Lorenzo, Juan I. Asensio-Perez, Yannis Dimitriadis, and Luis de-la-Fuente-Valentín. The Data Flow Problem in Learning Design: A Case Study. In *Proceedings of the International Conference on Networked Learning*, pages 285–292, Halkidiki, Greece, May 2008.

[117] Davinia Hernandez-Leo, Eloy Villasclaras-Fernandez, Juan I. Asensio-Perez, Yannis Dimitriadis, Miguel Bote-Lorenzo, and Jose Antonio Marcos-Garcia. Tuning IMS LD for Implementing a Collaborative Lifelong Learning Scenario. In *Sixth IEEE International Conference on Advanced Learning Technologies*, pages 1160–1161. IEEE, 2006.

[118] Eloy Villasclaras-Fernández, Juan I. Asensio-Pérez, Davinia Hernández-Leo, Yannis Dimitriadis, Luis de-la-Fuente-Valentín, and Alejandra Martínez-Monés. Implementing Computer-Interpretable CSCL Scripts with Embedded Assessment: A Pattern Based Design Approach. In *Techniques for Fostering Collaboration in Online Learning Communities: Theoretical and Practical Perspectives*, pages 261–277, 2010.

[119] Davinia Hernández Leo, Juan I. Asensio Pérez, and Yannis Dimitriadis. IMS Learning Design Support for the Formalization of Collaborative Learning Patterns. In *IEEE International Conference on Advanced Learning Technologies*, pages 350–354, Joensuu, Finlandia, August 2004.

[120] Mar Pérez-Sanagustín, Javier Burgos, Davinia Hernández-Leo, and Josep Blat. Considering the Intrinsic Constraints for Groups Management of TAPPS and Jigsaw CLFPs. In *2009 International Conference on Intelligent Networking and Collaborative Systems*, pages 317–322. IEEE, November 2009.

[121] Yongwu Miao, Kay Hoeksema, Heinz Ulrich Hoppe, and Andreas Harrer. CSCL scripts: modelling features and potential use. In *Proceedings of th 2005 conference on Computer support for collaborative learning*, pages 423–432, Taipei, Taiwan, 2005. International Society of the Learning Sciences.

[122] Francisco Jurado, Miguel Redondo, and Manuel Ortega. Specifying Collaborative Tasks of a CSCL Environment with IMS-LD. In *Cooperative Design, Visualization, and Engineering*, pages 311–317, 2006.

[123] Francisco Jurado, Ana I. Molina, William J. Giraldo, Miguel A. Redondo, and Manuel Ortega. Using CIAN for Specifying Collaborative Scripts in Learning Design. In *Proceedings of the 5th international conference on Cooperative Design, Visualization, and Engineering*, page 204, 2008.

[124] Florian König and Alexandros Paramythis. Towards Improved Support for Adaptive Collaboration Scripting in IMS LD. In *Sustaining TEL: From Innovation to Learning and Practice*, volume 6383 of *Lecture Notes in Computer Science*, pages 197–212. Springer Berlin / Heidelberg, 2010.

[125] Maha Khemaja, Sameh Ghallabi, and Valerie Monfort. Towards Intelligent Collaborative Learning Simulations: Extending the IMS LD Standard by Web Semantic Based ITSs. In *IEEE International Conference on Advanced Learning Technologies*, pages 642–643. IEEE Computer Society, 2010.

[126] Brendon Towle and Michael Halm. Designing Adaptive Learning Environments with Learning Design. In *Learning Design, A Handbook on Modelling and Delivering Networked Education and Training*, pages 215–226, 2005.

[127] Adriana Berlanga and Francisco Garcia. Modelling adaptive navigation support techniques using the IMS learning design specification. In *Proceedings of the sixteenth ACM conference on Hypertext and hypermedia*, pages 150, 148, Salzburg, Austria, 2005. ACM.

[128] Alexandra Cristea and Daniel Burgos. Authoring Adaptive Hypermedia and IMS Learning Design: A possible understanding? In *Sixth International Conference on Advanced Learning Technologies*, pages 1190–1191, Kerkrade, The Netherlands, 2006.

[129] Daniel Burgos and Marcus Specht. Adaptive e-Learning Methods and IMS Learning Design: An Integrated Approach. In *Sixth IEEE International Conference on Advanced Learning Technologies*, pages 1192–1193, July 2006.

[130] Daniel Burgos, Colin Tattersall, and Rob Koper. How to represent adaptation in e-learning with IMS - learning design. *Interactive Learning Environments*, 15(2):161–170, 2007.

[131] LN4LD. Units of Learning developed by several authors at Learning Network for Learning Design of The Open University of The Netherlands, 2005. Last visited November 2010 at http://moodle.learningnetworks.org/course/view.php?id=20.

[132] Telmo Zarraonandia, Paloma Díaz, Ignacio Aedo, Carmnino Fernández, and Juan Manuel Dodero. Evaluating the Runtime Adaptation of EML-Described Learning Processes. In *International Conference on Advanced Learning Technologies*, pages 413–417. IEEE, July 2008.

[133] Jesús G. Boticario and Olga C. Santos. An open IMS-based user modelling approach for developing adaptive learning management systems. *Journal of Interactive Media in Education*, 2, 2007.

[134] Sergio Gomez, David Huerva, Carolina Mejia, Silvia Baldiris, and Ramon Fabregat. Designing context-aware adaptive units of learning based on IMS-LD standard. In *EAEEIE Annual Conference*, pages 1–6. IEEE, June 2009.

[135] Jorge Hernandez, Silvia Baldiris, Olga C. Santos, Ramón Fabregat, and Jesus G. Boticario. Conditional IMS Learning Design Generation Using User Modeling and Planning Techniques. In *2009 Ninth IEEE International Conference on Advanced Learning Technologies*, pages 228–232. IEEE, July 2009.

[136] Valerie Monfort, Maha Khemaja, and Slimane Hammoudi. Extending the IMS LD standard with Adaptability. In *Advanced Learning Technologies, IEEE International Conference on*, volume 0, pages 320–322, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[137] IMS Global Learning Consortium. IMS Learning Design Best Practice and Implementation Guide, 2003. Last visited, November 2010 at http://www.imsglobal.org/learningdesign/ldv1p0/imsld_bestv1p0.html.

[138] John Loewen. *The study and application of IMS Learning Design specifications in a mobile context*. IEEE, December 2009.

[139] Imran A. Zualkernan, Hanan Al Darmaki, and Maha Shouman. A methodology for building simulation-based e-learning environments for Scrum. In *IEEE International Conference on Innovations in Information Technology*, pages 357–360, December 2008.

[140] Peter DeGrace and Leslie Hulet Stahl. *Wicked problems, righteous solutions*. Yourdon Press, 1990.

[141] Ricardo Amorim, Manuel Lama, and Eduardo Sanchez. Modelling and Implementation of the Astronomy Case Study with an IMS-LD Ontology. In *Sixth IEEE International Conference on Advanced Learning Technologies*, pages 1166–1167. IEEE, 2006.

[142] Eduardo Sanchez, Manuel Lama, Ricardo R. Amorim, Juan Carlos Vidal, and Adrian Novegil. On the use of an IMS LD ontology for creating and executing Units of Learning: Application to the Astronomy case study. *Journal of Interactive Media in Education*, 2008.

[143] Isabel S. Azevedo, Eurico Carrapatoso, and Carlos Vaz de Carvalho. Effective characterisation of learning objects. *International Journal of Advanced Media and Communication*, 2(3):236–246, 2008.

[144] Pierre Laforcade. In *Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007)*, pages 477–479. IEEE, July 2007.

[145] Onjira Sitthisak and Lester Gilbert. Extension of IMS LD for improved pedagogical expressiveness in assessment. In *Proceedings of the International Computer Assisted Assessment Conference*, 2010.

[146] Carlos Delgado Kloos, Abelardo Pardo, Mario Muñoz Organero, Mª Blanca Ibáñez, Raquel Crespo, Pedro Muñoz Merino, Luis de-la-Fuente-Valentín, Derick Leony, and Israel Gutiérrez. Some Research Questions and Results of UC3M in the eMadrid Excellence Network. In *EDUCON 2010 IEEE Annual Global Engineering Education Conference*, pages 1101–1110, Madrid, 2010.

[147] Anne Lejeune, Muriel Ney, Armin Weinberger, Margus Pedaste, Lars Bollen, Tasos Hovardas, Ulrich Hoppe, and Ton De Jong. Learning Activity Spaces: Towards Flexibility in Learning Design? In *International Conference on Advanced Learning Technologies*, pages 433–437, 2009.

[148] Ivan Martinez-Ortiz, Jose-Luis Sierra, and Baltasar Fernandez-Manjon. Authoring and Reengineering of IMS Learning Design Units of Learning. *IEEE Transactions on Learning Technologies*, 2(3):189–202, July 2009.

[149] IMS Global Learning Consortium. IMS Learner Information Package, 2005. Last visited, November 2010 at http://www.imsglobal.org/profiles/.

[150] IMS Global Learning Consortium. IMS Question and Test Interoperability Integration Guide, 2006. Last Visited, November 2010 at http://www.imsglobal.org/question/qtiv2p1pd2/imsqti_intgv2p1pd2.html.

[151] Yongwu Miao, Jo Boon, M Van Der Klink, Peter Sloep, and Rob Koper. Some Issues on Integrating IMS LD with IMS QTI. In *Support Interoperability and Reusability of Emerging Forms of Assessment*, 2009.

[152] Colin Tattersall, Daniel Burgos, Hubert Vogten, Harrie Martens, and Rob Koper. How to use IMS Learning Design and SCORM 2004 together. In *International Conference on SCORM*, Taipei, January 2006.

[153] Paul Sharples, Dai Griffiths, and Colin Tattersall. Integrating IMS Learning Design and ADL SCORM using CopperCore Service Integration. In *Service Oriented Approaches and Lifelong Competence Development Infrastructures Proceedings of the 2nd TENCompetence Open Workshop*, pages 84–91, Manchester, 2007.

[154] IMS Global Learning Consortium. IMS Tools Interoperability Guidelines, February 2006. Last Visited, November 2010 at http://www.imsglobal.org/ti/.

[155] Hubert Vogten, Harrie Martens, Rob Nadolski, Colin Tattersall, Peter van Rosmalen, and Rob Koper. CopperCore service integration. *Interactive Learning Environments*, 15(2):171–180, August 2007.

[156] Assessment Provision through Interoperable Segments, May 2004. Last visited, October 2010 at http://www.jisc.ac.uk/whatwedo/projects/apis.aspx.

[157] Pablo Moreno-Ger, Daniel Burgos, Jose-Luis Sierra, and Baltasar Fernández-Manjon. A Game-Based Adaptive Unit of Learning with IMS Learning Design and e-Adventure. *International Journal of Learning Technology*, 3(3):252 – 268, 2007.

[158] Pablo Moreno-Ger, Daniel Burgos, Iván Martínez-Ortiz, José Luis Sierra, and Baltasar Fernández-Manjón. Educational game design for online education. *Computers in Human Behavior*, 24(6):2530–2540, September 2008.

[159] Scott Wilson, Paul Sharples, and Dai Griffiths. Extending IMS Learning Design services using Widgets: Initial findings and proposed architecture. In *TENCompetence Open Workshop on Current research on IMS Learning Design and Lifelong Competence Development Infrastructures*, Barcelona, 2007.

[160] Paul Sharples, Dai Griffiths, and Scott Wilson. Using Widgets to Provide Portable Services for IMS Learning Design. In *Proceedings of the 5th International TENCompetence Open Workshop "Stimulating Personal Development and Knowledge Sharing"*, pages 57–60, Sofia, Bulgaria, 2008.

[161] Juan Manuel Dodero and Ernie Ghiglione. ReST-Based Web Access to Learning Design Services. *IEEE Transactions on Learning Technologies*, 1(3):190–195, July 2008.

[162] Rob Koper. Inscript: A courseware specification language. *Computers & Education*, 16(2):185–196, 1991.

[163] Luca Botturi, Michael Derntl, Eddy Boot, and Kathrin Figl. A classification framework for educational modeling languages in instructional design. In *6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006), Kerkrade (The Netherlands)*, pages 1216–1220, 2006.

[164] Luca Botturi and Todd Stubbs. *Handbook of Visual Languages for Instructional Design: Theories and Practices*. 2007.

[165] IMS Global Learning Consortium. IMS Simple Sequencing, 2003. Last Visited, November 2010 at http://www.imsglobal.org/simplesequencing/.

[166] Guillaume Durand and Stephen Downes. Toward Simple Learning Design 2.0. In *4th International Conference on Computer Science & Education*, pages 894–897, Nanning, China, July 2009. IEEE.

[167] Guillaume Durand, Luc Belliveau, and Benjamin Craig. Simple Learning Design 2.0. In *IEEE International Conference onAdvanced Learning Technologies*, pages 549–551, Los Alamitos, CA, USA, 2010. IEEE Computer Society.

[168] Manuel Caeiro-Rodriguez, Luis Anido-Rifon, and Martin Llamas-Nistal. A Proposal of Separation of Concerns in EMLs and Its Relation with LD. In *Sixth IEEE International Conference on Advanced Learning Technologies*, pages 76–80. IEEE, 2006.

[169] Manuel Caeiro-Rodríguez. poEML: A Separation of Concerns Proposal to Instructional Design. In *Handbook of Visual Languages for Instructional Design: Theories and Practices*, 2008.

[170] Manuel Caeiro-Rodriguez, Jorge Fontenla-González, Roberto Pérez-Rodríguez, and Luis Anido-Rifón. Instructional Design with PoEML in a E-learning-as-a-Service Model. Mixing Web and IPTV Learning Experiencies. pages 736–737, 2010.

[171] Christian Martel, Laurence Vignollet, Christine Ferraris, Jean-Pierre David, and Anne Lejeune. Modeling Collaborative Learning Activities on e-Learning Platforms. In *Sixth IEEE International Conference on Advanced Learning Technologies*, pages 707–709. IEEE, 2006.

[172] Pierre Laforcade. Towards a UML-Based Educational Modeling Language. In *Fifth IEEE International Conference on Advanced Learning Technologies (ICALT'05)*, pages 855–859. IEEE, 2005.

[173] Onjira Sitthisak and Lester Gilbert. Improving the pedagogical expressiveness of IMS LD. In *International conference on Technology Enhanced Learning Conference*, Taipei, 2009.

[174] Carlos Delgado Kloos, Abelardo Pardo, Mario Muñoz Organero, and Luis de-la-Fuente-Valentín. E-LANE: an e-learning initiative based on open source as a basis for sustainability. *International Journal of Continuing Engineering Education and Life-Long Learning*, 17(1):57 – 66, 2007.

[175] Walsh Norman and Richard L. Hamilton. *DocBook 5: The Definitive Guide*. O'Reilly Media, 2010.

[176] Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Generación de contenidos para plataformas educativas. In *Foro Hispano de .LRN*, Madrid, Spain, May 2005.

[177] M. Carmen Fernández Panadero, Mario Muñoz Organero, Luis de-la-Fuente-Valentín, and Carlos Delgado Kloos. Diseño, producción, impartición y evaluación de cursos a distancia mediante tecnologías y herramientas de código libre. In *Simposio Nacional de Tecnologías de la Información y las Comunicaciones en la Educación. Sexto Congreso Nacional de Informática Educativa*, pages 53–60, Granada, Spain, September 2005.

[178] The AOLServer platform, February 2007. Last visited, November 2010 at http://www.aolserver.com.

[179] tDOM Manual, 2003. www.tdom.org; Last visited, August 2008 at http://web.archive.org/web/20080805035611/http://www.tdom.org/.

[180] Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Change is Good. Improving Learning Design Flexibility at Run-time. In *EEE International Conference on Advanced Learning Technologies*, pages 1048–1050, Santander, Spain, July 2008.

[181] Eloy Villasclaras-Fernández, Davinia Hernández-Leo, Juan I. Asensio-Pérez, Yannis Dimitriadis, and Luis de-la-Fuente-Valentín. Interrelating assessment and flexibility in IMS-LD CSCL scripts. In *Scripted vs. Free CS Collaboration: Alternatives and Paths for Adaptable and Flexible CS Scripted Collaboration*, pages 39–43, 2009.

[182] Luis de-la-Fuente-Valentín, Mar Pérez-Sanagustín, Patricia Santos, Davinia Hernández-Leo, Abelardo Pardo, Carlos Delgado Kloos, and Josep Blat. System orchestration support for a flow of blended collaborative activities. In *2nd International Workshop on Adaptive Systems for Collaborative Learning*, Thessaloniki, Greece, 2010.

[183] Luis de-la-Fuente-Valentín, Julio Villena-Román, Abelardo Pardo, and Carlos Delgado Kloos. A cost-effective, scalable orchestration to promote continuous work in programming courses. Submitted to Computer Applications in Engineering Education.

[184] J. B. Arbaugh. Managing the on-line classroom: A study of technological and behavioral characteristics of web-based MBA courses. *The Journal of High Technology Management Research*, 13(2):203–223, 2002.

[185] Maia Dimitrova, Chris Sadler, Stylianos Hatzipanagos, and Alan Murphy. Addressing learner diversity by promoting flexibility in e-learning environments. In *Proceedings of the 14th International Workshop on Database and Expert Systems Applications, 2003.*, pages 287–291. IEEE, 2003.

[186] Frederick. Payne, John Ball, and Robert Snow. A corporate approach to the introduction of flexible delivery education. In *30th Annual Frontiers in Education Conference. Building on A Century of Progress in Engineering Education. Conference Proceedings*, pages F3D/1–F3D/5. Stripes Publishing, 2000.

[187] Tim S. Roberts. Flexible Learning: How Can We Get There from Here. In *In Proceedings of ASCILITE 2002*, pages 8–11, 2002.

[188] Marc Prensky. Digital Natives, Digital Immigrants Part 1. *On the Horizon*, 9(5):1–6, January 2001.

[189] Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Integrating generic services in GRAIL, the Learning Design Run-Time Environment in. LRN. In *LAMS European Conference*, pages 1–8, Cádiz, Spain, June 2008.

[190] Luis de-la-Fuente-Valentín, Yongwu Miao, and Abelardo Pardo. A Supporting Architecture for Generic Service Integration in IMS Learning Design. In *Times of Convergence. Technologies Across Learning Contexts, Lecture Notes in Computer Science (ECTEL-2008)*, pages 467–473, Maastricht, The Netherlands, 2008. Springer-Verlag.

[191] Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Using Third Party Services to Adapt Learning Material: A Case Study with Google Forms. In *Learning in the Synergy of Multiple Disciplines*, pages 744–750, Niza, October 2009. Springer Berlin / Heidelberg.

[192] Representation of dates and times, 1988.

[193] Luis de-la-Fuente-Valentín, Derick Leony, Abelardo Pardo, and Carlos Delgado Kloos. User identity issues in mashups for learning experiences using IMS Learning Design. *International Journal of Technology Enhanced Learning*, In Press, 2011.

[194] Jeroen J.G. van Merriënboer and John Sweller. Cognitive Load Theory and Complex Learning: Recent Developments and Future Directions. *Educational Psychology Review*, 17(2):147–177, 2005.

[195] OpenID Foundation. OpenID Specification, 2006. Last visited, January 2001 at http://openid.net/specs.bml.

[196] Eran Hammer-Lahav. RFC 5849: The OAuth 1.0 Protocol. Technical report, Internet Engineering Task Force, April 2010. Last Visited, January 2011 at http://tools.ietf.org/html/rfc5849.

[197] N. Asokan, Valtteri Niemi, and Kaisa Nyberg. Man-in-the-Middle in Tunnelled Authentication Protocols. In *Security Protocols*, pages 28–41. 2005.

[198] Luis von Ahn, Manuel Blum, Nicholas Hopper, and John Langford. CAPTCHA: Using Hard AI Problems for Security. In *Advances in Cryptology - EUROCRYPT 2003*, page 646. 2003.

[199] Edward F. Gehringer and W. Tyler Cross. A suite of Google services for daily course evaluation. In *IEEE Frontiers in Education Conference*, pages F4C–1–F4C–2. IEEE, October 2010.

[200] Inc. Google. Google Documents. Last visited, January 2011 at http://docs.google.com.

[201] Inc. Google. AuthSub for Web Applications. Authentication and Authorization for Google APIs. Last visited, November 2010 at http://code.google.com/intl/es-ES/apis/accounts/docs/AuthSub.html.

[202] Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Generic service integration in adaptive learning experiences using ims learning design. *Computers & Education*, 57(1):1160 – 1170, 2011.

[203] Rob Koper and Collin Tattersall. *Learning design: A handbook on modelling and delivering networked education and training*. Springer Berlin Heidelberg, 2005.

[204] MosaicLearning, December 2005. Last visited, December 2009 at http://mosaic.gast.it.uc3m.es/.

[205] NISE: National Institute for Science Education. Doing CL: CL Structures. Last viewed October 2007 from http://www.wcer.wisc.edu/archive/cl1/cl/, 1997.

[206] Davinia Hernández-Leo, Daniel Burgos, Colin Tattersall, and Rob Koper. Representing Computer-Supported Collaborative Learning macro-scripts using IMS Learning Design. In *Proceedings of the Second European Conference on Technology Enhanced Learning.*, Crete, Greece, 2007.

[207] Luis de-la-Fuente-Valentín. Complex collaborative models with a distant learning approach: an experience of use. Master's thesis, Carlos III University of Madrid, Leganes, February 2008.

[208] Luis de-la-Fuente-Valentín, Abelardo Pardo Sánchez, Carlos Delgado Kloos, Juan I. Asensio-Pérez, and Yannis Dimitriadis. Modelos de Aprendizaje Colaborativo en Entornos a Distancia con Learning Design: Un Caso de Estudio. *Revista Iberoamericana de Tecnologías del Aprendizaje*, 4(2):147–154, June 2009.

[209] Raquel Crespo, Abelardo Pardo, and Carlos Delgado Kloos. An adaptive strategy for peer review. In *34th Annual Frontiers in Education, 2004. FIE 2004.*, pages 765–771. IEEE, 2004.

[210] Angela Chow and Nancy Law. Measuring motivation in collaborative inquiry-based learning contexts. In *2005 Conference on Computer Support For Collaborative Learning: Learning*, pages 68–75, Taipei, Taiwan, May 2005.

[211] Davinia Hernández-Leo, Yannis Dimitriadis, and Juan I. Asensio-Perez. Computational Representation of Collaborative Learning Flow Patterns Using IMS Learning Desing. *Educational Technology & Society*, 8(4):75–89, February 2005.

[212] Jerry A. Colliver. Effectiveness of problem-based learning curricula: research and theory. *Academic medicine : journal of the Association of American Medical Colleges*, 75(3):259–66, March 2000.

[213] Armin Weinberger, Bernhard Ertl, Frank Fischer, and Heinz Mandl. Epistemic and social scripts in computer–supported collaborative learning. *Instructional Science*, 33(1):1–30, 2005.

[214] Fridolin Wild, Felix Mödritscher, and Steinn E. Sigurdarson. Designing for Change: Mash-Up Personal Learning Environments. *eLearning Papers*, 9, July 2008.

[215] Luis de-la-Fuente-Valentín, Derick Leony, Abelardo Pardo, and Carlos Delgado Kloos. Mashups in Learning Design: pushing the flexibility envelope. In *Proceedings of the First International Workshop on Mashup Personal Learning Environments*, Maastricht, September 2008.

[216] Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Experiences with GRAIL: Learning Design support in .LRN. In *Open Workshop on Current research on IMS-LD and Lifelong Competence Development Infrastructures*, Barcelona, Spain, June 2007.

[217] Tencompetence project, 2006. http://www.tencompetence.org; Last visited, May 2008 at http://web.archive.org/web/20080526085716/http://www.tencompetence.org/.

[218] Prolix project, 2005. Last visited, January 2011 at http://www.prolixproject.org.

[219] Dr. Robert E. Stake. *The Art Of Case Study Research*. Sage Publications, Inc, 1995.

[220] Peter Brusilovsky. Adaptive Hypermedia. *User Modeling and User-Adapted Interaction*, 11(1):87–110, 2001.

[221] Enrique Frias-Martinez, Sherry Y Chen, and Xiaohui Liu. Evaluation of a personalized digital library based on cognitive styles: Adaptivity vs. adaptability. *International Journal of Information Management*, 29(1):48–56, 2009.

[222] Sally He, Kinshuk, Hong Hong, and Ashok Patel. Granular approach to adaptivity in problem-based learning environment. In *Proceedings of IEEE International Conference on Advanced Learning Technologies*, pages 3–7, 2002.

[223] Richard Felder and Linda Silverman. Learning and Teaching Styles in Engineering Education. *Engineering Education*, 78(7):81, 674, 1988.

[224] María Del Puerto Paule Ruiz, M. Jesús Fernández Díaz, Francisco Ortín Soler, and Juan Ramón Pérez Pérez. Adaptation in current e-learning systems. *Computer Standards & Interfaces*, 30(1-2):62–70, January 2008.

[225] Cristina Hava Muntean. Improving learner quality of experience by content adaptation based on network conditions. *Computers in Human Behavior*, 24(4):1452–1472, July 2008.

[226] David J. Brown, David McHugh, Penny Standen, Lindsay Evett, Nick Shopland, and Steven Battersby. Designing Location based Learning Experiences for People with Intellectual Disabilities and Additional Sensory Impairments. *Computers & Education*, May 2010.

[227] Paul De Bra and Licia Calvi. AHA! An open Adaptive Hypermedia Architecture. *New Review of Hypermedia and Multimedia*, 4(1):115–139, 1998.

[228] John W. Creswell. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches (2nd Edition)*. Sage Publications, Inc, 2002.

[229] Robert Burke Johnson and Larry B. Christensen. *Educational Research: Quantitative, Qualitative, and Mixed Approaches*. Sage Publications, Inc, 2007.

[230] D.G. Rees. *Essential statistics, fourth edition*. Chapman and Hall/CRC, 2000.

[231] Michael Derntl, Susanne Neumann, Dai Griffiths, and Petra Oberhuemer. Investigating teachers ' understanding of IMS Learning Design : Yes they can! In *European Conference on Technology Enhanced Learning*, pages 62–77. Springer Verlag, 2010.

[232] ACM/AIS/IEEE. Computing Curricula 2005. The overview report. Technical report, September 2005.

[233] Garry L. White and Marcos P. Sivitanides. A theory of the relationships between cognitive requirements of computer programming languages and programmersćognitive characteristics. *Journal of Information Systems Education*, 13(1):59–66, 2002.

[234] Brenda Cheang, Andy Kurnia, Andrew Lim, and Wee-Chong Oon. On automated grading of programming assignments in an academic institution. *Computers & Education*, 41:121–131, September 2003.

[235] Dawn McKinney and Leo F Denton. Houston, we have a problem: there's a leak in the CS1 affective oxygen tank. In *SIGCSE '04: Proceedings of the 35th SIGCSE technical symposium on Computer science education*, pages 236–239, New York, NY, USA, 2004. ACM.

[236] Päivi. Kinnunen and Lauri Malmi. Why students drop out CS1 course? In *Proceedings of the second international workshop on Computing education research*, page 108. ACM, 2006.

[237] Susan Wiedenbeck. Factors affecting the success of non-majors in learning to program. In *Proceedings of the first international workshop on Computing education research*, ICER '05, pages 13–24, New York, NY, USA, 2005. ACM.

[238] M. A. Albanese and S. Mitchell. Problem-based learning: a review of literature on its outcomes and implementation issues. *Academic medicine : journal of the Association of American Medical Colleges*, 68(1):52–81, January 1993.

[239] Kenneth A. Reek. A software infrastructure to support introductory computer science courses. *ACM SIGCSE Bulletin*, 28(1):125–129, 1996.

[240] James M. Feldman and James Jones. Semiautomatic testing of student software under Unix(R). *IEEE Transactions on Education*, 40(2):158–161, May 1997.

[241] Simone C. dos Santos, Maria da Conceição Moraes Batista, Ana Paula C. Cavalcanti, Jones O. Albuquerque, and Silvio R.L. Meira. Applying PBL in Software Engineering Education. In *Proceedings of the 2009 22nd Conference on Software Engineering Education and Training*, pages 182–189. IEEE Computer Society, 2009.

[242] Jianping Zhang and Haiqiong Du. The PBL's Application Research on Prolog Language's Instuction. In *2008 International Workshop on Education Technology and Training & 2008 International Workshop on Geoscience and Remote Sensing*, pages 112–114. IEEE, December 2008.

[243] Kuo-Hung Tseng, Feng-Kuang Chiang, and Wen-Hua Hsu. Interactive processes and learning attitudes in a web-based problem-based learning (PBL) platform. *Computers in Human Behavior*, 24(3):940–955, May 2008.

[244] Esko Nuutila, Seppo Törmä, Päivi Kinnunen, and Lauri Malmi. *Learning Programming with the PBL Method—Experiences on PBL Cases and Tutoring*, pages 47–67. Springer, 2008.

[245] Utku Köse. A web based system for project-based learning activities in "web design and programming" course. *Procedia - Social and Behavioral Sciences*, 2(2):1174–1184, 2010.

[246] Dick Grune and Matty Huntjens. Detecting copied submissions in computer science workshops. Technical report, Informatica Faculteit Wiskunde & Informatica, Vrije Universiteit, Amsterdam, 1989.

[247] Mar Pérez-Sanagustín, Gustavo Ramírez-González, Davinia Hernández-Leo, Mario Muñoz Organero, Patricia Santos, Josep Blat, and Carlos Delgado Kloos. Discovering the Campus Together: a mobile and computer-based learning experience. *Journal of Network and Computer Applications*, In Press, 2011.

[248] Mario Munoz-Organero, Gustavo A. Ramíez-González, Pedro J. Munoz-Merino, and Carlos Delgado Kloos. A Collaborative Recommender System Based on Space-Time Similarities. *IEEE Pervasive Computing*, 9(3):81–87, July 2010.

[249] Tony Navarrete, Patricia Santos, Davinia Hernández-Leo, and Josep Blat. QTIMaps: A model to enable web-maps in assessment. *Submitted*.

[250] Derick Leony, Luis de-la-Fuente-Valentín, Abelardo Pardo, and Carlos Delgado Kloos. Adaptación de material educativo guiada por ims learning design: experiencias con .LRN. *Revista Iberoamericana de Educación a Distancia*, 13(2):209–235, 2010.

Appendix A

GSI API methods

It follows a list of the methods that must be provided by a GSI service adapter. The GSI layer will call these methods when the IMS LD engine needs the interaction with the third-party tool.

Methods used before the service adapter is selected

get_support_url

Returns an URL that links to the documentation of the service adapter. Such documentation will be right after uploading the UoL in the IMS LD player and will let to select the best matching tool for the uploaded UoL.

Methods used during the service adapter configuration

get_zero_act_urls

Returns a list of all the web pages that will be used during the *act zero*. Such pages are provided by the service adapter and will depend on the requisites of the supported third-party tool.

Each element of the returned list should contain the following elements:

- The URL of the resource to be accessed
- The title of the link that will be placed in the activity tree
- If they exists, the learning objects associated to the resource. These learning object will be accesible from the *environment* of the activity.

The *act zero* will have an *activity structure* that contains the URLs in the list.

get_zero_act_roles

Returns the list of IMS LD roles that must take part in the activities of the *act zero*. It is suggested for the method to follow these steps:

1. Obtain a list of the UoL roles that will use the third-party tool (GSI groups)
2. Check the permissions that these roles will have in the third-party tool.

3. Decide, according to the permissions and the requirements of the supported third-party tool, which role will take part in the *act zero*.

The IMS LD player will create a *role-part* for each of the roles in the returned list. The *role-part* will relate these roles with the *activity-structure* created with the URLs returned by `get_zero_act_urls`.

Methods used during course instantiation

initialize_user

If the service adapter keeps a record of its users, this method is used to initialize the user data in the internal database.

Explicit user initialization is required to prevent recording innecessary data.

request_configured_instance

This method receives a *user-id*. The method will check the user's permissions in the third-party tool and, depending on such permissions and on the functionality of the supported third-party tool, instantiates an instance and return the URL (or list of URLs) that will be used to access such instance.

Each returned URL should contain the following elements:

- The URL of the resource to be accessed
- The title of the link that will be placed in the activity tree

get_zero_act_visibility

This method says who will have access to the activities in the *act zero*. That is, several roles might be involved in the *act zero* (see `get_zero_act_roles`), but it does not mean that they have to perform the same tasks. Actually, the different roles will usually access to different tasks.

This method returns a list of the activities that will perform each role that participates in the *act zero*. The IMS LD player will change the visibility of the activities according to the list returned by this method.

Methods called during the enactment phase

zero_act_scripts

This method decides if a tcl script must be executed as reponse to the completion of an activity of the *act zero*. The decision is based on who completed the activity and which activity was completed.

Then, the method is called by the IMS LD player whenever an activity of the *act zero* is terminated. The method receives the following information:

- The activity that has been terminated
- The user who has performed the activity
- A boolean indicating if the method will receive future calls from the same course instance. That is, if all the users have finished the *act zero*.

action_list_execute

Receives a list of actions that should be executed in the third-party tool and ask the tool to execute them. This method is called whenever a IMS LD event triggers a GSI action. The actions are defined in the manifest.

It should be indicated which user has triggered the actions, i.e. which user is calling the action in the third-party tool.

perform_startup_actions

It is the same than the `action_list_execute` method. The difference is that `perform_startup_actions` is called when the third-party tool is being instantiated and the actions are executed for all the users in the course, not only for a particular user.

get_external_value

This method is called whenever the IMS LD player requests information from the third-party tool. This method is in charge of requesting such information and translating it to a GSI understandable format.

In most cases, the retrieved information regards a particular user. Thus, the method receives the *user-id* of the person that owns the data. The authentication token of the user is taken from the information gathered in the *act zero*.

Third-party tools usually offer their information in a proprietary format. The `get_external_value` method must parse the such information and return the expected value, which is defined by the array type (*contribution*, *context*, *custom*. See the description of the GSI vocabulary for more information).

Appendix B

Questionnaires for the analysis and evaluation of the *MOSAIC* experience

The questionnaires offered to authors, tutors and students in the experience described in Section 6.3 are presented here. The questionnaires are in Spanish, that is the language in which they were presented to the participants.

Cuestionario Dirigido a los Diseñadores de la UoL de la Experiencia Mosaic sobre la Especificación del Flujo de Artefactos

Instrucciones : Este cuestionario está dirigido al diseñador de la UoL de la experiencia Mosaic. El cuestionario se enfoca exclusivamente a la especificación del flujo de artefactos.

Cuestionario Como autor del diseño de la UoL del caso Mosaic:

¿Cuál era tu experiencia previa en el diseño de UoL's con IMS-LD al momento de realizar este diseño?
Respuesta [0-nada ... 6-mucha]:

¿Qué problema(s) tuviste, si hubo alguno, al momento de especificar el flujo de artefactos?
Respuesta Abierta:

¿Crees que la aproximación que usaste para especificar el flujo de artefactos es propensa de errores para el autor del diseño? Explica.
Respuesta Abierta:

¿Crees que la aproximación que usaste para especificar el flujo de artefactos te produjo una carga cognitiva adicional a la de especificar por ejemplo, el flujo de actividades?

Respuesta Abierta:

¿Cómo compararías la especificación que usaste para el flujo de artefactos, respecto de la especificación que usaste para el flujo de actividades?
Respuesta Abierta:

Preferirías que el flujo de artefactos se especificara de forma continua, de manera análoga a la especificación del flujo de actividades?
Respuesta [Si,No, Da Igual]:

¿Cuál es tu valoración general sobre la aproximación usada en la experiencia para especificar el flujo de artefactos?
Respuesta [0-pobre ... 6-buena]:
Comentarios:
Respuesta Abierta:

¿Por qué no usaste una aproximación basada en especificar artefactos como propiedades para el flujo de artefactos?
Respuesta Abierta:

Cuestionario Dirigido a los Administradores y Tutores de la Experiencia Mosaic sobre la Gesión de Ficheros por parte de los Usuarios

Instrucciones : Este cuestionario está dirigido a los tutores y administradores de la experiencia Mosaic. El cuestionario se enfoca exclusivamente a la gestión de artefactos por parte de los usuarios.

Cuestionario Como tutor y/o administrador del caso Mosaic:

¿En qué grado los usuarios cumplían las instrucciones de la actividad para almacenar sus ficheros?
Respuesta [0-no cumplían ... 6-cumplían]:
Respuesta Abierta:

¿Podías distinguir entre un usuario que no subía su fichero del que lo subía a otra carpeta o con otro nombre? Explica.
Respuesta Abierta:

¿Crees que esta aproximación usada para gestionar ficheros es propensa de errores por parte del usuario? Explica.
Respuesta Abierta:

¿Crees que esta aproximación usada para gestionar ficheros le agrega una carga cognitiva adicional a los usuarios? Explica.
Respuesta Abierta:

¿Cuál es tu valoración general sobre la aproximación usada en la experiencia para la gestión de ficheros por parte del usuario?
Respuesta [0-pobre ... 6-buena]:
Comentarios:

¿Qué recomendaciones harías para mejorar la gestión de ficheros por parte del usuario?
Respuesta Abierta:

Cuestionario Dirigido a Usuarios sobre la Gestión de Ficheros en la Experiencia Mosaic

Instrucciones : Este cuestionario es anónimo y está dirigido a todas las personas que participaron en la experiencia Mosaic en el rol de estudiantes. El cuestionario se enfoca exclusivamente a la experimentación del usuario en el tema de gestión de ficheros durante la experiencia. Las preguntas son cerradas, sin embrago tu comentarios son muy importantes para este estudio.

Contexto : Si recuerdas, la aproximación usada en la experiencia Mosaic para la gestión de ficheros, era un repositorio de ficheros compartido que era gestionado por cada usuario siguiendo las instrucciones de la actividad:

¿Almacenaste alguna vez un fichero en una carpeta distinta o con un nombre distinto al indicado en las instrucciones de la actividad?
Respuesta [Si, No]:
Comentarios:

¿Si tu respuesta a la pregunta anterior fue "Sí", ¿cuál fue la razón de no seguir las instrucciones de la actividad para salvar tu fichero? (a) Entendí mal las instrucciones (b) Apliqué mal las instrucciones (c) Olvidé las instrucciones (d) Ignoré las instrucciones (f) Otra (Indicar Cuál)
Respuesta[Inciso]:
Comentarios:

¿Tuvistes alguna vez problemas para localizar los ficheros de alguno de tus compañeros durante una actividad peer-review?
Respuesta [Sí, No]:
Comentarios:

Contesta a esta pregunta si has contestado "Sí" a la pregunta anterior. Si en una actividad peer-review, el fichero de tu compañero no se encontraba en la carpeta correspondiente, ¿eras consciente de que posiblemente tu compañero lo hubiera subido a otra carpeta o lo hubiera salvado con otro nombre?
Respuesta [Si, No]:
Comentarios:

¿Crees que esta aproximación para almacenar y acceder a los ficheros es propensa de errores por parte de los usuarios?, ¿en qué medida?

Respuesta 1 [Sí, No]:
Respuesta 2 [0-poco ... 6-mucho]:
Comentarios:

¿Crees que te producía una carga cognitiva adicional el tener que seguir las instrucciones de la actividad para saber donde almacenar tu fichero y con qué nombre, o donde acceder a los ficheros de tus compañeros y con qué nombres?, ¿en qué medida?
Respuesta 1 [Sí, No]:
Respuesta 2 [0-nada ... 6-mucha]:
Comentarios:

Si limitamos la responsabilidad del usuario a subir ficheros de su sistema local al sistema gestor y viceversa, sin tener que preocuparse dónde los almacena el sistema ni de cómo les llama, de forma que el sistema gestor recupere automáticamente los ficheros que le corresponden a cada usuario en una actividad. ¿Crees que se reduciría la carga cognitiva del usuario?, ¿en qué medida?
Respuesta 1 [Sí, No]:
Respuesta 2 [0-poco...6-mucho]
Comentarios:

¿Cuál es tu valoración general sobre la aproximación usada en la experiencia Mosaic sobre la gestión de ficheros por parte del usuario?
Respuesta [0-mala...6-buena]:
Comentarios:

Appendix C

Questionnaires for the analysis and evaluation of the *Science Week* experience

The questionnaires offered to the participants of the *Semana de la Ciencia* workshop, described in Section 7.2 are presented here. The questionnaires are in Spanish, that is the language in which they were presented to the participants.

Cuestionario previo

Tu nombre
Respuesta texto libre

¿Cuál es tu experiencia previa en el uso de ordenadores?
1=Ninguna, 2 = Poca, 3 = Normal, 4 = Me manejo con soltura, 5 = Soy un experto

Incluye comentario con respecto a tu experiencia con los ordenadores
Respuesta texto libre

¿A qué te dedicas?
Respuesta texto libre

¿Cuál es tu motivación para participar en este taller?
Respuesta texto libre

Cuestionario intermedio

Tu nombre
Indica el mismo nombre que pusiste en el cuestionario previo

Valora de 1 a 5 tu experiencia previa con la adaptación de materiales. Considera también los ejemplos en los que no hubiera ordenadores implicados en la adaptación

(1 = Ninguna experiencia; 5 = Mucha experiencia)

Teniendo en cuenta únicamente la adaptación realizada por ordenador, valora tu experiencia previa con la adaptación de materiales.
(1 = Ninguna experiencia; 5 = Mucha experiencia)

Indica la utilidad que consideres para los tipos de adaptación

- *Basada en el perfil de alumno. (1 = Poco útil; 5 = Muy útil)*
- *Basada en la actividad del alumno. (1 = Poco útil; 5 = Muy útil)*
- *Basada en el escenario de trabajo. (1 = Poco útil; 5 = Muy útil)*

¿Qué efecto crees que tiene este la adaptación de material en el proceso educativo?
(1 = Muy negativo; 5 = Muy positivo)

Valora de 1 a 5 la relación dificultad/beneficio de la adaptación de materiales.
(1 = Demasiado difícil, no compensa el benefício; 5 = Muy beneficioso, el trabajo queda compensado)

¿Te ha resultado comprensible el método de trabajo seguido en el ejemplo?
(1 = Nada comprensible; 5 = Fácilmente comprensible)

¿Crees que es un ejemplo realista? o dicho de otra forma, ¿Cómo de cercana está la adaptación del ejemplo a tu entorno de trabajo/educativo?
(1 = Nada cercana; 5 = Muy cercana)

Valora la utilidad del ejemplo en un caso práctico real
(1 = Nada útil; 5 = Muy útil)

Valora de 1 a 5 la relación dificultad/beneficio en el ejemplo
(1 = Demasiado difícil, no compensa el benefício; 5 = Muy beneficioso, el trabajo queda compensado)

Comentarios sobre las posibilidades que te sugiere este ejemplo.
Respuesta texto libre

Cuestionario final

Tu nombre
Indica el mismo nombre que pusiste en el cuestionario previo

Una vez que has creado tu propio material, valora de 1 a 5 la relación dificultad/beneficio del esquema de trabajo planteado
(1 = Demasiado difícil, no compensa el benefício; 5 = Muy beneficioso, el trabajo queda compensado)

Valora de 1 a 5 la facilidad que has tenido para...

- *Incluir en la plantilla el material del curso. (1 = Nada fácil, 5 = muy fácil)*
- *Crear un cuestionario. (1 = Nada fácil, 5 = muy fácil)*
- *Interactuar con la hoja de cálculo. (1 = Nada fácil, 5 = muy fácil)*
- *Incluir reglas de adaptación en el curso. (1 = Nada fácil, 5 = muy fácil)*
- *Hacer una previsualización del curso. (1 = Nada fácil, 5 = muy fácil)*
- *En general, para crear tu propio curso adaptativo . (1 = Nada fácil, 5 = muy fácil)*

Valora la facilidad de implantar esta técnica en tu entorno de trabajo. La pregunta no se refiere a dificultades administrativas, sino a la acogida por parte de los docentes.
(1 = Nada fácil, 5 = Muy fácil)

¿Crees que el uso de una hoja de cálculo para gestionar las repuestas aporta algún valor añadido al curso?
(1 = No, en absoluto, 5 = Si, mucho)

Explica cuál es este valor añadido.
Respuesta texto libre

Nombra otras aplicaciones de internet cuyo uso crees que beneficia la impartición de un curso.
Indica también el contexto en el que usarías dicha aplicación.
Respuesta texto libre

Valora de 1 a 5 tu impresión de...

- *Contenido del taller. (1 = No me ha gustado, 5 = Me ha gustado mucho)*
- *Dinámica del taller. (1 = No me ha gustado, 5 = Me ha gustado mucho)*
- *Materiales ofrecidos. (1 = No me ha gustado, 5 = Me ha gustado mucho)*
- *Duración del taller. (1 = No me ha gustado, 5 = Me ha gustado mucho)*

Comentarios generales del taller
Respuesta texto libre

Appendix D

Questionnaires for the analysis and evaluation of the experience in *System Programming*

The questionnaires offered to the students in the experience described in Section 7.3 are presented here. The questionnaires are in Spanish, that is the language in which they were presented to the participants.

Cuestionario para profesores

Tareas del profesor

¿Cuál es tu rol en la asignatura?
Profesor de prácticas/teoría

Experiencia docente
Indica tu experiencia en tareas docentes, tanto en esta asignatura como en cualquier otro tipo de experiencia docente. Indica también tu experiencia con sistemas de gestión de aprendizaje (Moodle y similares).
Respuesta de texto libre

Indica qué medios has utilizado para acceder al enunciado del proyecto

- *Los enunciados "en crudo" en el repositorio de contenidos de profesores*
- *El rol de profesor en la plataforma de despliegue de proyecto*
- *El rol de alumno ficticio en la plataforma de despliegue del proyecto (sólo disponible en el proyecto 2)*
- *Otro (indica cuál)*

Indica qué medios has utilizado para acceder a los recursos asociados al proyecto
(código fuente, página de entrega, resultados de la competición...)

- *Los enunciados "en crudo" en el repositorio de contenidos de profesores*

- *El rol de profesor en la plataforma de despliegue de proyecto*
- *El rol de alumno ficticio en la plataforma de despliegue del proyecto (sólo disponible en el proyecto 2)*
- *Otro (indica cuál)*

Indica qué medios has utilizado para acceder a la información relativa a los alumnos
Es decir, tiempos de entrega, código entregado, etc.

- *La cabina de mando del reproductor de cursos*
- *El enlace de descarga de código*
- *Los logs "en crudo" ofrecidos por la plataforma alojada en plato*
- *Otri (indica cuál)*

¿Has necesitado algún recurso para tutelar o evaluar a los alumnos pero no has podido acceder a él?
Indica qué recurso y por qué no has podido acceder
Respuesta de texto libre

Para el proyecto 1, valora numéricamente el grado de integración que has percibido a la hora de acceder a los recursos de tutela/evalución que has necesitado. Por ejemplo en un sistema bien integrado todos los elementos necesarios para el trabajo están relacionados entre sí y se accede a ellos de forma similar
5=Todos los elementos muy bien integrados, 1= Elementos excesivamente distribuídos

Para el proyecto 2 valora numéricamente el grado de integración que has percibido a la hora de acceder a los recursos de tutela/evalución que has necesitado. Por ejemplo en un sistema bien integrado todos los elementos necesarios para el trabajo están relacionados entre sí y se accede a ellos de forma similar
5=Todos los elementos muy bien integrados, 1= Elementos excesivamente distribuídos

Explica tus respuestas a las 2 preguntas previas
Respuesta de texto libre

Percepción del alumnado

¿A qué nivel crees que se ha conseguido el objetivo de que los alumnos lleven a cabo un trabajo sostenido?
5 = Trabajo muy sostenido con el tiempo, 1 = Trabajo muy poco sostenido

En relación a la pregunta anterior ¿en qué fuentes de información basas tu respuesta?
Respuesta de texto libre

¿Crees que la carga de trabajo impuesta a los alumnos por el proyecto es coherente con la planificación de la asignatura?

5 = Muy coherente, 1 = Muy poco coherente

Si crees que la carga de trabajo no es coherente explicalo
Respuesta de texto libre

¿Crees que la dificultad del proyecto es coherente con la planificación de la asignatura?
5 = Muy coherente, 1 = Muy poco coherente

Si crees que la dificultad no es coherente explicalo
Respuesta de texto libre

Cómo cuantificarías el interés en la práctica por parte del alumnado:
5 = Mucho interés, 1 = Poco interés

¿Cuáles crees que son los factores de influencia de dicho interés?
Respuesta de texto libre

El interés mostrado por los alumnos a lo largo del proyecto ¿ha influído en tu motivación a la hora de afrontar tus tareas como docente?
5 = Mucha influencia, 1 = Poca influencia

Explica tu respuesta anterior
Respuesta de texto libre

Crees que el planteamiento global de los proyectos ha supuesto un beneficio para los alumnos.
5=Mucho beneficio, 1= Ningún beneficio

Matiza tu respuesta anterior
Respuesta de texto libre

Flujo de actividades

Valora el impacto que ha tenido la corrección semiautomática de entregas de proyecto sobre:
5 = Muy beneficioso, 1 = Nada beneficioso

- Las habilidades de programación adquiridas por los alumnos
- La calidad de las entregas
- El ritmo de trabajo del alumnado

Valora el impacto que crees que ha tenido el acceso secuencial por parte de los alumnos a las distintas partes del enunciado sobre:
5 = Muy beneficioso, 1 = Nada beneficioso

- Las habilidades de programación adquiridas por los alumnos
- La calidad de las entregas
- El ritmo de trabajo del alumnado

Valora el impacto que crees que ha tenido la imposibilidad de reentregar un código ya aceptado como válido sobre los siguientes aspectos:
5 = Muy beneficioso, 1 = Nada beneficioso

- Las habilidades de programación adquiridas por los alumnos
- La calidad de las entregas
- El ritmo de trabajo del alumnado

Con respecto a las penalizaciones propuestas en caso de no cumplir los plazos previstos valora numéricamente el impacto que ha tenido sobre:
5 = Muy beneficioso, 1 = Nada beneficioso

- Las habilidades de programación adquiridas por los alumnos
- La calidad de las entregas
- El ritmo de trabajo del alumnado

Añade los comentarios que consideres oportunos con respecto a las 4 preguntas previas
Respuesta de texto libre

Carga de trabajo del profesorado y valoración global

¿Que factores (si ha habido alguno) han contribuido a reducir la carga de trabajo del profesorado en el proyecto?
Respuesta de texto libre

¿Qué factores (si ha habido alguno) han contribuido a incrementar la carga de trabajo del profesorado en el proyecto?
Respuesta de texto libre

Valora numéricamente la carga de trabajo del profesorado a lo largo del proyecto
5=Mucho más de lo necesario, 1=Mucho menos de lo necesario

Independientemente de si la carga de trabajo ha sido alta o baja ¿Consideras que se adecúa a lo que se espera por parte del profesorado?
5=Sí, mucho, 1=No, nada

Explica tu respuesta a las dos preguntas anteriores
Respuesta de texto libre

¿Estás de acuerdo con el planteamiento general del proyecto?
5=Muy de acuerdo, 1=Nada de acuerdo

Explica tu respuesta a la pregunta anterior
Respuesta de texto libre

Tomando como base la experiencia del presente curso ¿consideras adecuado volver a utilizar el mismo planteamiento en cursos futuros?
Respuesta de texto libre

Cuestionario para alumnos

Instrucciones

Esta encuesta es opcional (no es obligatorio hacerla), no evaluada (no se tiene en cuenta la nota final) y anónima (no se sabe a quién pertenece cada respuesta) El objetivo de esta encuesta es mejorar, de cara a cursos futuros, el planteamiento de los proyectos realizados a lo largo de la asignatura. No hace falta que contestes a todas las preguntas, lo que sí que es necesario es que pulses el botón "Submit" en la última página del cuestionario. Gracias por tu colaboración

Carga de trabajo

Cuántas horas de trabajo has dedicado en total al primer proyecto
La pregunta se refiere al trabajo individual llevado a cabo, no a l suma de los integrantes de la pareja.

- Menos de 10 horas
- Entre 10 y 15 horas
- Entre 15 y 20 horas
- Entre 20 y 25 horas
- Más de 25 horas

El tiempo dedicado al primer proyecto, ¿ha estado concentrado en días concretos, o distribuído a lo largo de la duración del proyecto?
5 = Trabajo muy sostenido, 1 = Trabajo muy concentrado en momentos puntuales

Consideras que este tiempo de trabajo es, en relación con los créditos que vale la asignatura y al porcentaje de nota del proyecto:
5 = Muy poco tiempo de trabajo, 1 = Excesivo tiempo de trabajo

Valora numéricamente la dificultad que, según tu opinión, ha tenido el primer proyecto
5 = Muy fácil, 1 = Muy difícil

¿Crees que esta dificultad es acorde a lo que cabe esperar con respecto a los contenidos de la asignatura?
La pregunta se refiere a la dificultad de sacar adelante las prácticas, no del tiempo dedicado a ello.

- No, es más difícil de lo que debería
- Sí, es acorde a lo esperable
- No, es más fácil de lo que debería

¿Consideras apropiados los criterios de evaluación del proyecto 1?
5 = Muy apropiados, 1 = Nada apropiados

¿Crees que la nota que has sacado en el primer proyecto es justa?
Incluye los comentarios con respecto a la evaluación que consideres apropiados

Respuesta tipo texto libre.

Cuántas horas de trabajo has dedicado en total al segundo proyecto
La pregunta se refiere al trabajo individual llevado a cabo, no a la suma de los integrantes de la pareja.

- Menos de 10 horas
- Entre 10 y 15 horas
- Entre 15 y 20 horas
- Entre 20 y 25 horas
- Más de 25 horas

El tiempo dedicado al segundo proyecto, ¿ha estado concentrado en días concretos, o distribuído a lo largo de la duración del proyecto?
5 = Trabajo muy sostenido, 1 = Trabajo muy concentrado en momentos puntuales

Consideras que este tiempo de trabajo es, en relación con los créditos que vale la asignatura y al porcentaje de nota del proyecto:
5 = Muy poco tiempo de trabajo, 1 = Excesivo tiempo de trabajo

Valora numéricamente la dificultad que, según tu opinión, ha tenido el segundo proyecto
5 = Muy fácil, 1 = Muy difícil

¿Crees que esta dificultad es acorde a lo que cabe esperar con respecto a los contenidos de la asignatura?
La pregunta se refiere a la dificultad de sacar adelante las prácticas, no del tiempo dedicado a ello.

- No, es más difícil de lo que debería
- Sí, es acorde a lo esperable
- No, es más fácil de lo que debería

¿Consideras apropiados los criterios de evaluación del proyecto 2?
5 = Muy apropiados, 1 = Nada apropiados

¿Crees que la nota que has sacado en el primer proyecto es justa?
Incluye los comentarios con respecto a la evaluación que consideres apropiados
Respuesta tipo texto libre.

Hay una errata en la redacción de la pregunta, aquí se ha reproducido tal cual. El enunciado debería preguntar por el proyecto 2, lo que era fácilmente deducible por el contexto. A tenor de los resultados, creo que los alumnos lo han interpretado bien, pero queda la duda.

Sobre las preguntas anteriores, añade los comentarios que consideres apropiados.
Respuesta tipo texto libre.

Sobre el sistema de entregas

¿Crees que el método de acceso a los enunciados, recursos y sistema de entregas ha sido suficientemente claro?
5 = Muy claro, 1 = Poco claro

Complementa tu respuesta a la pregunta anterior
Respuesta tipo texto libre.

El acceso secuencial a las diferentes entregas (no disponer del siguiente enunciado hasta no haber terminado el actual), ha resultado beneficioso o perjudicial para el desarrollo del código.
5 = Muy Beneficioso, 3 = Neutro, no ha influído, 1 = Muy perjudicial

Complementa tu respuesta a la pregunta anterior
Respuesta tipo texto libre.

La información ofrecida por el servidor tras realizar la corrección automática, ¿Te ha supuesto alguna ayuda para encontrar problemas en tu código?
5 = Sí, mucho, 1 = No, nada

¿Consideras que el uso de este sistema de entregas te ha supuesto un coste en tiempo adicional?
5 = No, nada, 1 = Sí, ha supuesto un alto coste.

¿Ha influído el sistema de entregas en tu forma de trabajo?
5 = Sí, mucho, 1 = No, nada

El sistema de entregas, basado en la corrección automática del código, ¿te ha resultado beneficioso o perjudicial a la hora de realizar el proyecto?
5 = Muy Beneficioso, 3 = Neutro, no ha influído, 1 = Muy perjudicial

Complementa tu respuesta a las 4 preguntas previas con los comentarios que consideres oportunos
Respuesta tipo texto libre.

El sistema de entregas planteado para los proyectos (varias entregas por proyecto, con corrección automática) ¿Crees que ha tenido algún impacto en la nota que has obtenido?
5 = Sí, mucho; 1 = No, no ha influído

Complementa tu respuesta a la pregunta anterior
Respuesta tipo texto libre.

¿Has tenido algún problema para determinar cuál de los enunciados es el que te indica qué hacer en la siguiente actividad?
5 = No, el enunciado ha sido suficientemente explícito, 1 = Sí, por momentos no tenía claro qué enunciado me correspondía

¿Has tenido algún tipo de problema para encontrar/descargar los recursos (código fuente) ofrecidos como complemento a los enunciados?

5 = No, ningún problema, 1 = Sí, muchos problemas.

Complementa tu respuesta a las 2 preguntas previas con los comentarios que consideres oportunos
Respuesta tipo texto libre.

¿Has necesitado, o te hubiese gustado saber a través del sistema de entregas cuál es el estado de las entregas de tus compañeros?

- Sí
- No
- No me importa

¿Te hubiese gustado o importado que tus compañeros conociesen el estado de tus entregas?

- Sí
- No
- No me importa

Teniendo en cuenta lo anterior, ¿te gustaría volver a utilizar el mismo sistema de entregas en el futuro?
En asignaturas de cursos posteriores, o en la misma asignatura si repites.

- Sí
- No
- NS/NC

Comenta tu respuesta a la pregunta anterior
Respuesta tipo texto libre.

Sistema de competición

El código que has entregado, ¿ha formado parte de la competición?

- Si
- No
- Other:

¿Has obtenido puntuación extra por la competición?

- Si
- No

¿Has realizado algún esfuerzo extra de desarrollo de código para que tu código obtenga buena puntuación en la competición?

- Si

- No

En caso de haber respondido afirmativamente a la pregunta anterior, indica en qué ha consistido el esfuerzo extra
Respuesta tipo texto libre.

¿Cómo definirías el premio (la nota) conseguido por la competición?
Elige una de las siguientes opciones

- Excesivamente alto
- Un poco alto
- Adecuado
- Un poco bajo
- Excesivamente bajo

¿Dirías que la competición ha supuesto un aliciente en tu motivación a la hora de realizar el proyecto?
5 = Sí, me ha motivado mucho, 1 = No, no me motivado nada

Incluye cualquier comentario con respecto al sistema de competición que consideres adecuado
¿Qué cambiarías?
Respuesta tipo texto libre.

Otros temas

¿Tienes aprobada la asignatura de programación del primer cuatrimestre de primero?

- Sí
- No
- Other:

Valora la disponibilidad del profesorado a lo largo de la asignatura
5 = Muy buena disponibilidad, 1 = Muy mala disponibilidad

Valora la actitud del profesorado con respecto a los alumnos
5 = Actitud muy positiva, 1 = Actitud muy negativa

Valora la calidad de las explicaciones dadas por el profesorado a lo largo de la asignatura
5 = Muy buenas explicaciones, 1 = Muy malas explicaciones

Danos tu opinión sobre el profesorado de la asignatura
Respuesta tipo texto libre.

Danos tu opinión general sobre la asignatura de Programación de Sistemas.
Respuesta tipo texto libre.

Appendix E

Questionnaires for the analysis and evaluation of the *Meeting the campus* experience

The questionnaires offered to students in the experience described in Section 7.4 are presented here. The questionnaires are in Spanish, that is the language in which they were presented to the participants.

Cuestionario para alumnos

Di tres de los aspectos que hayas aprendido sobre el campus que consideres importantes para tu primer año como estudiante de ingeniería.
Respuesta texto libre

Indica 3 aspectos en que creas que te ha ayudado la experiencia *Meeting the Campus*
Respuesta texto libre

¿La experiencia te ha servido para conocer a gente que no conocías?
Si/No

Valora del 1 al 5 cuánto crees que te ha ayudado la experiencia *Descobreix el Campus de la Comunicació* para conocer el Campus y sus servicios.
1, Muy poco; 5 = Mucho

Valora del 1 al 5 la medida en que el trabajo en grupo te ha ayudado a reforzar tus ideas y conocimientos sobre el Campus.
Respuesta texto libre

Valora del 1 a 5 la medida en que la elección de los miembros del grupo te ha resultado adecuada considerando la actividad realizada.
Respuesta texto libre

Justifica tu respuesta a la pregunta anterior.
Respuesta texto libre

¿Qué crees que tenías en común con los miembros de tu grupo?
Respuesta texto libre

Valora del 1 al 5 en qué medida crees que la tecnología móvil ha sido adecuada para la exploración del campus.
1, Muy poco adecuada; 5 = Muy adecuada

Valora entre 1 (Muy fácil) y 5 (Muy difícil)

1. *Identificar mi actividad en cada momento*
2. *Avanzar en la secuencia de actividades*
3. *Entender qué tengo que hacer*

¿Has requerido la ayuda del profesor para entender qué tocaba hacer en cada momento?
Si / No

Valora entre 1 y 5 en qué medida están integradas las diferentes actividades.
(1, es un conjunto de actividades y tecnologías inconexas; 5, las actividades y tecnologías estás muy relacionadas entre sí)

¿Crees que las pausas entre actividades rompen el ritmo de la actividad?
(1,no, no rompen el ritmo; 5, sí, sí que rompen el ritmo)

Añade los comentarios que creas oportunos sobre las 5 preguntas previas.
Respuesta texto libre

Si tuvieras que escoger entre una de las tres fases de la actividad cuál escogerías?

- *Exploración Web*
- *Exploración con móviles*
- *Actividad en grupo*

Justifica tu respuesta:
Respuesta texto libre

¿Crees que la combinación de las tres actividades te aporta más que realizar las actividades por separado?
Si / No
¿Por qué? *Respuesta texto libre*

¿Recomendarías la experiencia a tus compañeros que no han podido realizarla? ¿Por qué?
Respuesta texto libre

¿Repetirías la experiencia? ¿Por qué?
Respuesta texto libre

¿Te ha gustado que el test final se tuviera que contestar a través del ordenador?
Si / No / Indiferente

Valora del 1 al 5 qué opinas de las preguntas que integraban Google Maps?

- *Me han gustado*
- *Me han sorprendido*
- *Las he encontrado útiles en el contexto de la experiencia*

¿Tienes alguna sugerencia sobre la actividad para próximas ediciones?
Respuesta texto libre