

# Open Source and IMS Learning Design: Building the Infrastructure for eLearning

David Griffiths  
Universitat Pompeu Fabra  
Barcelona, Spain  
david.griffiths@upf.edu

Ray Elferink  
RayCom BV, Netherlands  
Raymond@raycom.com

Josep Blat  
Universitat Pompeu Fabra  
Barcelona, Spain  
josep.blat@upf.edu

Sara Zondergeld  
RayCom BV, Netherlands  
sara@raycom.com

**Abstract** – The development of open, flexible eLearning specifications has significant implications for and interactions with the FOSS movement. A short overview of eLearning specifications is provided, focusing on the difference between SCORM and Learning Design (LD). The significance of LD for FOSS is examined, and common values identified. The particular contribution made by FOSS to LD infrastructure is discussed, and the importance of reference applications described. An overview is given of the FOSS applications available, divided into design time and run time, with particular reference to LD editors and the CopperCore Learning Design engine.

## I. INTRODUCTION

This paper is strongly informed by discussions held in the context of two European projects in which the authors are involved. Firstly they are members of the Special Interest Group on Organisation and Management Issues of SIGOSSEE/JOIN [1] which promotes the use of Free and Open Source Software in education. Secondly, the UNFOLD project [2], which supports the adoption of IMS Learning Design, which we explain later. The cross fertilisation between the two communities of participants in the projects has given rise to many of observations made here, and provided the context for examining the contributions which Open Source and Open Standards can bring to each other communities of developers and users.

Free and Open Source Software (FOSS) has made significant progress in European education in recent years [3] and more recently the Creative Commons [4] initiative has extended this to open content. In a separate development, Open eLearning Specifications for interoperability have been established, which provide the means whereby eLearning resources can be exchanged between systems. We outline the growth of Open eLearning Specifications, and focus in particular on IMS Learning Design (LD), in part because it is the focus of a substantial current open source effort, and also because it has features which are of particular relevance to FOSS.

In the discussion below we discuss both the relevance of LD for FOSS eLearning implementations, and the particular ways in which FOSS can support the development of LD software infrastructure. We then provide an overview of the applications available so far and those under development. Unless otherwise stated, all the applications discussed are FOSS.

## II. THE RELEVANCE OF IMS LEARNING DESIGN FOR FOSS ELEARNING IMPLEMENTATIONS

### A. An outline of open eLearning specifications

A key initial milestone in Open eLearning Specifications was the Ariadne project in 1997 [5], which worked on defining metadata for the identification of learning objects. The Ariadne metadata itself included the more general Dublin Core Metadata Initiative metadata for electronic resources from 1994[6]. In 1997 IMS Global Learning Consortium Inc. (IMS) was established to produce specifications for all aspects of distributed learning [7]. IMS has become the leading force in defining Open eLearning specifications, and has adopted much of the Ariadne's metadata in the IMS Learning Object Metadata (LOM) specification. IMS has produced a growing suite of specifications, some of which have achieved high levels of adoption. A number of these, such as Content Packaging, Question and Test Interoperability, and Simple Sequencing, are incorporated in the Sharable Content Object Reference Model (SCORM) produced by Advanced Distributed Learning (ADL) [8]. This process of consolidation, and the wide adoption of the SCORM mean that there is now a solid base of accepted de facto standards for eLearning interoperability.

It may be worth stressing that open source code does in itself imply interoperability. While FOSS provides many advantages, it does not automatically mean that documents can be ported to other systems, or that different systems can work together. Interoperability specifications provide important added value, especially in educational environments which are largely heterogeneous.

The choice of an open eLearning specification is not merely a technical issue, it has strong consequences for the educational activities which are supported, as argued by Friesen [9]. In the case of SCORM it is well positioned to address the needs of a single learner in programmed learning<sup>1</sup>, but cannot handle multiple users, and cannot represent the role of the teacher. The pedagogy which it supports may be characterised as an implementation of the "conduit" metaphor, as described by Lakoff [11]. It should be stressed that this is not because the specifications which make up the SCORM are in themselves bad. On the contrary they are an essential part of the infrastructure for interoperable eLearning. The problem is that they are insufficient, because they deal primarily with educational content, without supporting flexible activities and collaboration.

### B. The significance of IMS Learning Design for FOSS

A more recent IMS specification, Learning Design (LD)

<sup>1</sup>Programmed learning: "Learning in which the students progress at their own rate using workbooks, textbooks or electromagnetic resources that provide information in discrete steps, test learning at each step and provide immediate feedback about achievement"[10]

sets out meet this lack, and like IMS LOM it also builds on previous European work, in this case Educational Modelling Language (EML) from the Open University of the Netherlands. LD provides a language which can model pedagogic scenarios, including multiple learners in collaboration, and the role of the teacher. It does this by providing a precise description of how people in roles carry out activities with learning resources. For more information on LD, please see [7, 12, 13]. The specification was published in 2003, and work on creating the tools needed to work with it is currently reaching fruition.

It is clear that FOSS can be used to support all kinds of eLearning, including programmed learning. Nevertheless, there is a substantial overlap between the values of the FOSS community and those of educators who work with pedagogies which may broadly be described as constructivist. Among other aspects, this tradition emphasises the importance of collaboration, of discourse, of multiple valid viewpoints, and the idea that each learner needs to be supported in constructing their own meaning with culturally appropriate tools. This clearly has much in common with the FOSS communities stress on collaboration, adaptation of software to local requirements, and localisation to many cultures. FOSS eLearning using open specifications and informed by a constructivist approach was not possible prior to the publication of LD, and consequently many FOSS developers in education had to choose between two undesirable options: lack of interoperability, or restriction to a constrained (and perhaps unsympathetic) pedagogic framework. LD resolves this conflict, and will therefore be welcomed by many FOSS developers.

To be fully effective, however, a specification for interoperability needs to be adopted in both FOSS and proprietary software. The SCORM has achieved high levels of adoption, in part because it has been supported by received substantial direct financial support from the US Department of Defence, totalling 84.4 million dollars between 2003 and 2009 [14], plus mandated compliance from Federal authorities. Much of this funding has gone to subsidise the production of proprietary applications. So far LD has not received anything near such funding or mandated support, despite the fact that the specification is much more complex, and hence implementation more challenging. The need for LD to succeed is felt most keenly by learners and teachers and educational institutions, rather than by publishers and software companies. Consequently it is not prudent to rely on proprietary software providers to create critical mass for LD, even though they are not opposed to it. To overcome this strong bias towards the dominance of the SCORM there is a need for a concerted effort to create a complete FOSS infrastructure for the LD, and this is indeed coming about, funded by European, national and institutional sources, as we describe below.

### III THE IMPORTANCE OF FOSS IN CREATING AN INFRASTRUCTURE FOR LD

The implementation of the LD specification, which has been coordinated by the Valkenburg Group (with a significant FOSS presence). The Group has developed a reference architecture which defines the applications which need to be built, and has, together with the UNFOLD

project, coordinated the development process. There are other higher level structures which provide the context for the reference architecture, including the eLearning Framework (ELF) based in the UK, and the OKI and SAKAI initiatives in the USA, all of which have a strong FOSS orientation. For a discussion of the Valkenburg Group architecture, and these frameworks and their relationship to LD, see Wilson [15]. Service based architectures have also been addressed in the SBLDS project [16] funded by JISC in the UK.

Thus there is collaborative framework for development of an infrastructure for open specifications which promises to lead to a complete FOSS infrastructure for eLearning, equivalent to that available for the Internet. This is very valuable as a unifying structure for the development efforts of FOSS developers in education, who can be sure that their work will be interoperable and adaptable for a world wide community of users. Communities such as UNFOLD provide a central store of information so that developers know what has been achieved so far in creating an LD infrastructure, and can identify the most urgent needs. They also welcome FOSS developers with a forum where they can provide input into the evolving architecture for LD.

The use FOSS in creating this infrastructure makes a contribution to LD (and the wider frameworks) in three important respects. Firstly it offers a way to achieve critical mass. Potential adopters need to be shown the benefits of LD before they will adopt it, and publishers and proprietary software developers want to be shown that there is a market before they will develop for it. FOSS can provide the impetus to drive adoption by making free tools available which potential users can try out at no cost. Secondly, the LD specification is extensive and complex, and it is far from simple to implement software to edit compliant documents, and servers on which the resulting XML can be run. The only way to ensure interoperability is to have reference implementations which represent the agreed interpretation of the specification and ways to implement it, and which have their source code open to inspection by other developers. If these are not available, then each team of developers will make their own decisions when faced with a problem of interpretation of the specification. The sum of the variant interpretations in the different sets of applications processing LD documents in different installations would lead to inconsistent output being provided to learners (or perhaps even a failure to run) and interoperability would be lost.

Thirdly, the well defined architecture for LD development, and the coordination of development, means that developers can often build on existing FOSS implementations in order to add new functionality. This greatly speeds implementations of the specification, and makes adoption more likely.

### IV. PROGRESS TO DATE IN DEVELOPING A FOSS INFRASTRUCTURE FOR LD

The types of tools required for working with LD are discussed in Griffiths [17], and we direct readers there for a detailed discussion. Here we limit ourselves to providing an overview of the specialised tools required, and a summary of the principal FOSS projects underway at the time of writing. The tools for working with LD may be divided into two main categories, design time and run time,

which we will examine in turn. The most significant developments are summarised at a table in following section, together with URLs.

1. *Design time tools.* These are the various categories of editors, together with compliance testing applications. We do not consider the enabling framework within which these operate. LD Units of Learning<sup>2</sup> (UOLs) are encoded as XML files, and a valid UOL can be written in any text editor (if the author has sufficient skill and patience!). There is, however, no reason why an author should ever see the raw XML [18], which should be handled transparently by the application. The first generation of LD editors to appear has represented the UOL as a branching tree, with an interface which enables the author to navigate through the tree and enter the appropriate values for the LD elements. An editor of this type at a minimum helps the author by hiding the complexity of the XML syntax, and by guiding them through the hierarchy, so that elements are not misplaced. There are, however, other important functionalities which tree based editors can provide. They can handle the internal references which need to be updated whenever a new resource is added or changed. It is very useful if they provide a mechanism for the user to incorporate existing fragments of UOLs (for example an activity structure) and incorporate them in a new UOL. It may also be valuable to be able to rename the elements, to make them more understandable to members of particular communities of groups of users, or for different language groups.

Examples of FOSS implementations of tree based editors include RELOAD [19], aL.Fanet LD Editor [20], COSMOS [21], and CopperAuthor [22]. They may divide the specification up into sections, as does RELOAD, providing separate tabs for editing roles, environments activities and method. These implementations are leading the way in LD implementation, as no proprietary editors have yet been released.

Authoring a UOL in a text editor is a job for a programmer, and tree based editors greatly simplify the task so that it can be undertaken by anyone who is, for example, comfortable authoring web sites, and who is willing to put in the effort to understand the structure of a UOL and the purpose of the elements which make it up. This means that they are appropriate for specialists in the development of learning materials and online courses, but they are still too complex and extensive for teachers (or learners) to be able to handle, as they are not able to invest the necessary time. It is important that teachers can engage with the authoring process so that, for example, they can add and change resources in existing UOLs, and inspect a UOL and recognise if it is appropriate for their purposes. It is also the case that some teachers want to be able to understand and control the computing environment in which they are working, and may want to set up their own courses. For this to be possible the complexity of the specification has to be reduced in some way. This can be achieved in a tree based editor by constraining the options available to the author, so that many design decisions are taken in advance and hidden from the author. A template of this sort can also be presented in many other ways, for example the EduPlone [23] LD authoring facility offers a form based interface for

the creation of a restricted set of simple UOLs. This approach may be more effective if combined with a patterns based analysis of the pedagogical problems which the templates address. An encouraging development is that the Moodle [24] community is showing interest in LD, and there is no doubt that an LD compliant version of the Moodle system would be a very valuable addition to the available LD infrastructure.

Another approach is to provide users with predefined chunks of UOLs which they can combine to form valid UOLs. These may be patterns (structures which resolve a specified pedagogic problem) or primitives (which are commonly used components which teachers can combine for their own purposes) See Griffiths [25] for a discussion of these terms and their implications for LD. These components need not be limited to a single LD element, and could consist of, for example, a combination of a role part and a service. This is done in the ASK-LDT editor, produced by the CERTH Centre [26], where the author can drag predefined structures into the UOL as it is being constructed. ASK-LDT is not intended as a tool for teachers, but does provide an example of the kind of functionality which could be provided. A tool which is specifically intended for teachers is LAMS (Learning Activity Management System) [27] which provides an easy to use interface enabling authors to drag activities into a sequence. This has so far not been LD compliant, but an LD import/export capability is scheduled for release in June 2005.

FOSS authoring tools for teachers are scarce, and as yet not mature. In this the development of LD infrastructure is following that of, for example, the relatively simple HTML specification, where it took some years before editors appeared which could be used by non-experts.

A high level interface such as that provided by LAMS is not only useful to interfaces intended for teachers. The MOT+ editor [28] (a proprietary application) is an editor for learning designers which uses a graphical editor to create courses following the MISA design method. The resulting designs can be exported to LD. This means that learning designers can use tools optimised for the methods which they prefer, and maintain interoperability. The DialogPlus toolkit takes a similar approach, enabling authors navigate through a pedagogic taxonomy (which does not follow the structure of LD). At present the development team are working on exporting to LD the pedagogic structures defined in DialogPlus.

Authors also need to validate their UOLs, to be confident that they are fully compliant. Some editors ensure that only valid UOLs can be created, and the CopperCore Learning Design Engine (see next section) also performs validation.

2. *Runtime tools.* An LD player application accepts a UOL as an XML file, and interprets it to provide learners with the appropriate resources, services and activities as they work. This is not a straightforward process. Firstly, information has to be added before learning can commence. A UOL is an abstract representation of a pedagogic structure, and in simple terms it may be considered an interoperable lesson plan. Each time a cohort of learners use a UOL for learning this is called a run. Before a UOL can be run information has to be added about the specific learners and teachers who will be involved, dates may need to be specified, and services may

---

<sup>2</sup> A Unit of Learning is a defined term in the LD specification, giving a precise meaning to the broad unit of learning concept as an independent, relatively self-contained piece of learning.

need to be set up. It is assumed that much of this will be done automatically, reading from databases which maintain this information in other parts of the providing institution, but at the operation can also be carried out using the Clicc application, produced by the Open University of the Netherlands and distributed with CopperCore (see below). This has a command line interface, making the learning curve for using it rather steep, but a new interface is under development.

Secondly, once a specific run is populated with users and other necessary information, the player application has to keep track of states of all the learners as they evolve, and provide the appropriate resources and activities over time. Implementing such a system is a major task, and the OUNL has made a substantial contribution to player development by providing the CopperCore Learning Design Engine. This application handles all the underlying processing in the complex core of the player, but provides only a simple user interface. It is intended as a tool for developers which enables them to build on the engine and focus on providing innovative interfaces for players. CopperCore is also a reference implementation of a player engine (as discussed above), and provides a guide for later implementers who want to know how certain aspects of the specification should be interpreted.

Work has also been carried out towards wrapping CopperCore in a service layer, in the SBLDS Project (Service Based Learning Design System) funded by JISC (Joint Information Systems Committee) in the UK, opening the way to a range of new applications.

One particular kind of player which is required is a viewer for authors, so that they can preview a UOL with dummy users as they are working on it. If this is not available then the UOL has to be populated with users before it can be loaded into a player such as CopperCore. The RELOAD team have provided a player of this sort [18], which should perhaps be more correctly termed a viewer.

Another runtime application which will be required is a repository of UOLs. Any learning materials repository can be used, but it would be useful to add specific LD features to help users identify the most appropriate UOL for their purposes, using graphic representations and/or controlled vocabularies. The problem here is largely one of understanding what those representations and vocabularies should be, and the answer to this can only come through practice. It is therefore not surprising that these features have not yet been implemented, but as the technical implementation is not especially challenging this should not hold up completion of the infrastructure.

## V. CONCLUSIONS

The most significant FOSS applications for LD currently available or under development are as follows. The web addresses are as of April 12 2005.

### 1. Currently available

#### Editors

- Reload
- aL.Fanet LD Editor
- CopperAuthor

#### LD Player Engine

- CopperCore

#### Players

- CopperCore has a simple player incorporated
- RELOAD viewer

#### Tool for populating UOLs

- Clicc (included in CopperCore)

## 2. Under development

#### Editors

- DialogPlus
- ASK LDT
- COSMOS

#### Editor / Player

- LAMS

#### Players

- SBLDS Service wrapping for CopperCore
- Alfabet player

The FOSS infrastructure for eLearning described above is a huge enterprise, and it will not be possible to assess the final results for a number of years in the future. The infrastructure for Learning Design has, however, reached the point where use of the specifications is a viable option, with the critical open source applications already in place. The key targets for future development are clear, based on the architectures established by the Valkenburg Group, and there is an active community working on applications. The wider FOSS framework initiatives such as ELF, OKI and SAKAI encourage us to believe that this technology is becoming embedded at a strategic level, and that the emerging FOSS infrastructure for LD will be part of an overarching FOSS infrastructure for eLearning. There remain a number of needs to be met which would facilitate adoption of LD. In particular easier to use high level authoring environments and templates need to be developed, more varied and sophisticated player interfaces provided, together with repositories with specific LD features. The provision of applications which ease the administration of LD systems and their integration with enterprise systems in education institutions would also be very advantageous.

## VI. REFERENCES

- [1] SIGOSSEE / JOIN, OSSITE, *project website*. Available at <http://www.ossite.org>
- [2] UNFOLD, *project website*. Available at <http://www.unfold-project.net>
- [3] Vuorikari, R., *Why Europe Needs Free and Open Source Software and Content in Schools*. 2004, European Schoolnet. p. 10. Available at [http://www.eun.org/insight-pdf/special\\_reports/Why\\_Europe\\_needs\\_foss\\_Insight\\_2004.pdf](http://www.eun.org/insight-pdf/special_reports/Why_Europe_needs_foss_Insight_2004.pdf)
- [4] Creative Commons, *website*. accessed.10th April 2005, available at <http://creativecommons.org/>
- [5] The Ariadne Foundation, *The two ARIADNE projects*. Available at <http://www.ariadne-eu.org/en/foundation/history.html>
- [6] Dublin Core Metadata Initiative, *History of the Dublin Core Metadata Initiative*. 2005. Available at <http://dublincore.org/about/history/>
- [7] IMS Global Learning Consortium Inc., *corporate website*. Available at <http://www.imsglobal.org/>
- [8] ADL *Corporate Website*. Available at

<http://www.adnet.org/>

- [9] Friesen, N., *Three Objections to Learning Objects*, in *Online Education Using Learning Objects*, R. McGreal, Editor. 2004, Routledge/Falmer: London.
- [10] UNESCO, *Thesaurus*. Available at <http://databases.unesco.org/thesaurus/>
- [11] Lakoff, G. and Johnson, M., *Metaphors We Live By*. 1980, Chicago and London: University of Chicago Press.
- [12] UNFOLD, *project website*. accessed.10th April 2005. Available at [www.unfold-project.net](http://www.unfold-project.net)
- [13] Koper, R. and Tattersall, C., eds. *Learning Design: modelling and implementing network-based education & training*. 2005, Springer. 412.
- [14] US Department of Defence, *UNCLASSIFIED FISCAL (FY) 2005 DESCRIPTIVE SUMMARIES*, available at <http://www.dtic.mil/descriptivesum/Y2005/DHRA/0603769SE.pdf>
- [15] Wilson, S., *Architectures to Support Authoring and Content Management with Learning Design*, in *Learning Design, a Handbook on Modelling and Delivering Networked Education and Training*, R. Koper, Editor. 2005, Springer. p. 41-62.
- [16] JISC, *SBLDS project*. Available at <http://www.jisc.ac.uk/index.cfm?name=sbls>
- [17] Griffiths, D., et al., *Learning Design Tools*, in *Learning Design: modelling and implementing network-based education & training*, R. Koper and C. Tattersall, Editors. 2005, Springer Verlag. p. 109-135.
- [18] Olivier, B., *The Learning Design Specification*, in *Learning Design, a Handbook on Modelling and Delivering Networked Education and Training*, R. Koper, Editor. 2005. p. 21-40.
- [19] Reload, *Project website*. Available at <http://www.reload.ac.uk/>
- [20] aL.Fanet, *Project website*. Available at <http://alfanet.ia.uned.es/>
- [21] **COSMOS site**.
- [22] CopperAuthor, SourceForge site. Available at <http://sourceforge.net/projects/copperauthor/>
- [23] EduPlone, *corporate website*. Available at [http://eduplone.net/index\\_html?cl=en](http://eduplone.net/index_html?cl=en)
- [24] Moodle, *coporate website*. Available at <http://moodle.org>
- [25] Griffiths, D. and Blat, J., *The Role of Teachers in Editing and Authoring Units of Learning using IMS Learning Design*. International Journal on Advanced Technology for Learning, 2005. **2**(3).
- [26] CERTH, *ASK Website*. Available at <http://www.iti.gr/db.php/en/pages/ASK.html>
- [27] LAMS International, *corporate website*. Available at <http://www.lamsinternational.com>
- [28] Technologies Cogigraph Inc. *MOT Knowledge Editor*. Available at [http://www.cogigraph.com:90/cogigraph/article.php3?id\\_article=19](http://www.cogigraph.com:90/cogigraph/article.php3?id_article=19)