Implementing Learning Design to support web-based learning.

Patrick Mcandrew, Will Woods, Alex Little, Martin Weller, Rob Koper, Hubert Vogten

Implementing Learning Design to support web-based learning

P. McAndrew*, W.I.S. Woods*, A. Little*, M.J. Weller*, Rob Koper**,Hubert Vogten**
* Intstitute of Educational Technology, Open University, Walton Hall,
Milton Keynes, MK7 6AA. UK.
** Educational Technology Expertise Centre (OTEC), Open Universiteit Nederland,
Valkenburgerweg 177, 6419 AT Heerlen, NL.

Abstract

In this paper we consider an initial implementation of a system for managing and using IMS Learning Design (LD) to represent online learning activities. LD has been suggested (Koper & Olivier, 2004) as a flexible way to represent and encode learning materials, especially suited to online and web-based learning while neutral to the pedagogy that is being applied. As such it offers a chance to address a gap in the preparation of learning materials and their eventual use by students by providing a formal description of the approach, roles and services needed for a particular unit of learning. The potential in learning design that most interests us is its scope for the exchange of validated and formalised designs and so encouraging reuse. Until full implementations exist this potential cannot be explored and it is hard to predict if learning design will provide value in describing either full courses or in describing isolated activities. The initial work is therefore to implement a system for managing, validating and inspecting learning design building on collaboration between the Institute of Educational Technology at the Open University UK (OUUK) and the Educational Technology Expertise Centre (OTEC) at the Open University of the Netherlands (OUNL), who produced a Learning Design Engine CopperCore (http://coppercore.org/) released under Open Source.

Introduction

The need to develop methods for the representation and sharing of effective learning methods and materials raises many challenges (McAndrew, Brasher & Hardy, 2004). We take a position that a valuable contribution can be made by enabling accurate representation of different designs for learning and allowing their modification and exchange. It is worth noting though that there has been previous enthusiasm for building up an "Educational Object Economy" and that in practice such an economy has proved hard to support. Previous research (Laurillard & McAndrew, 2002) identified aspects of the barrier to success to lie in the granularity at which objects could be described and the ownership that attached to the different levels of the original design, particular instance and reinterpretations. These aspects remain important to investigate and the approach of learning design appears to offer both a possible solution to the need to represent the designs and also a possible tool to explore models of how to validate and share the designs. Once we have a practical instantiation we are then seeking to consider ways to analyse the designs and link them to the learning theories they implement. Example research questions are:

- Is it possible to analyses designs before they are applied them? (For example, construct an activity analysis using an Activity Theory approach (Engeström, 1987)).
- Are there patterns within designs that are needed to support particular approaches? For example if a social-constructivist approach is used, it might be that explicit collaboration and peer feedback must appear in the design.
- Are designs stable? Minor variations in design may lead to very different learner experiences.

A distinction is worth making between "learning design" (in lower case letters) for the idea of capturing different designs and the IMS specification for Learning Design, in this paper that will be represented by the initials LD.

IMS Learning Design (LD) is described in detail in the documentation produced to accompany it as part of the suite of IMS specifications and standards (IMSLD, 2003), so is only described briefly here. It is a development of the Educational Modelling Language (EML) (Hummel et al, 2003) designed by the OUNL to enable flexible representation of the elements within online courses; not just the materials but also the order in which activities take place, the roles that people undertake, key criteria for progression, and the services needed for presentation to learners. The learning design specification does not detail how the course material itself is represented but rather how to package up the overall information into a structure that is modelled on a play, with acts, roles (actors) and resources. For those familiar with IMS and its application in VLEs this means that LD can take the place of the simpler Content Package but with more meaning. An XML schema and best practice guidelines are provided and there is scope to represent complete courses using LD and deliver them through "players", this would then is similar to the approach taken by the OUNL where courses described in EML are mapped onto sets of students and tutors and delivered through the EduBox system.

LD has clearly reinvigorated interest in standardised approaches to online learning and discussion has reflected dissatisfaction with the more monolithic approaches inherent in using most VLEs. Partly though this interest comes from what may be possible if the designs behind learning do become formalised and exchangeable (i.e. the possibilities of a learning design approach in general, rather than the IMS LD specification itself). Thus some work is current on learning design "inspired" approaches rather than systems that follow the specification in detail. Notable impact has been achieved by the Learning Activity Management System (LAMS) (Dalziel, 2003) system which offers a very easy to use interface to creating a simplified sequence of sub-activities with a fixed choice of roles and limited control structures. The initial implementation provides a complete solution from specifying the activity, selecting cohorts of students, offering them tools designed to fit with those activities, and monitoring their progress as a group. The LAMS system follows the overall approach of learning design but avoids the rigour of using generic representations of roles and services to represent the resulting activity according to the LD specification. The strength and engagement of LAMS needs to be valued while recognising the limitations in the pilot system (of course these may well be addressed as the system itself develops). At present the limited set of roles quickly becomes a problem in representing existing activities, rather than modifying exemplar activities originally designed in LAMS, and the requirement to use built-in tools currently rules out its use in large-scale established environments such as are used in the OUUK.

While LD gives a way to represent a complete course it was apparent at a meeting of those interested in learning design (an informal commercial and university interest group formed of key players in e-learning) that a key driver for enthusiasm for the new specification is the opportunity it can give for more abstract representations and the capture of designs as templates for reuse. In our investigation we need to see if there is a viable way to produce abstractions of learning designs, possibly as a way to then produce further instantiations just as LAMS enables variations in the original activity patterns that fed into the building of the tool. For example, it is possible to envisage a "learning design wizard" that produces full specifications based on variations in particular design. The results of this can then be formal learning designs, typically produced as XML representations, ready to be validated and instantiated using a LD player based on the existing LD engine.

CopperCore

The CopperCore LD engine has the architecture shown in figure 1. In this architecture three application programme interfaces (APIs) are provided. The first provides access to validation; the second to services; and, the third provides the presentation aspects.

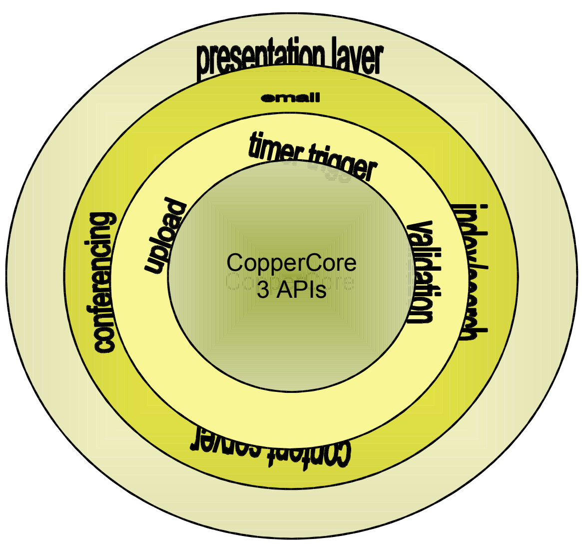

Figure 1: CopperCore interfaces and integrations

CopperCore operates on XML documents that follow the LD specification, it applies validation against the appropriate schema and also provides further semantic validation of the attributes within the design. Using the prototype engine has allowed the team at the OUNL to refine some of the early examples of LD and shows the importance of working with implementations of the specifications. All the APIs are exposed using XML and can be called or manipulated externally to provide file management and student management, this allows separation of the content and services from the engine itself. The final layer that is needed in a player system is interpretation for presentation, in the prototype implementation this is through XSL style sheets applied to the transformed learning design. The core implementation uses Java to provide the combining and integration code, the work at the OU is to use the API to produce Java handlers for the particular designs and services we want to investigate. In particular we aim to use learning design within an existing knowledge management environment (the Knowledge Network) that is built using ColdFusion and Microsoft SQLServer. Using this environment enables us to take a very flexible and experimental approach to providing the services needed for designs to work, it will also help build experience towards using either a commercial VLE for services or a component based approach to services.

LD summary

Learning design is based on the analogy of a play, with roles and acts. The IMS Learning Design Best Practice and Implementation Guide states that the core concept of the Learning Design Specification "is that regardless of pedagogical approach, a *person* gets a *role* in the teaching-learning process, typically a *learner* or a *staff* role. In this role he or she works toward certain *outcomes* by performing more or less structured *learning* and/or *support activities* within an *environment*. The environment consists of the appropriate *learning objects and services* to be used during the performance of the activities. Which role gets which activities at what moment in the process, is determined by the *method* or by a *notification*."

It thus aims to be descriptive, or expressive, not prescriptive.

There are three levels of LD:

- Level A – this is the simplest form and covers the activities, roles, acts and environment used in a learning design
- Level B – this introduces the notion of properties and conditions. It is at this level that LD becomes useful, as it allows what-if conditions and storing properties (such as student performance) to allow for multiple paths through learning material
- Level C – this supports notification or messaging between system components, which allows for a more dynamic workflow and personalisation.

An LD example

To make the use of learning design more understandable we will introduce a particular example. The example has been chosen to be simple and involve only one actor (the learner) however it is important to realise that some of the power of LD is in its ability to extend to collaborative tasks. It has also been restricted to those aspects that are part of level A of LD; this still allows very rich structures but without explicit storing of progress, decision points (level B) or reference to other systems through messages or notification (level C).

The model for the example comes from a task designed by one of the authors for a Master's level course produced by the Open University and supported through the UK eUniversities platform (Weller et al, 2003). In this task each student is asked to read a section, place their own views on a two dimensional grid, then reflect on the positions of other authors and place those views on the same grid. The LD approach described in the best practice guide suggests first completing a template "use case narrative" describing the design, then mapping it into a UML representation before producing a document conforming to the XML schema for LD.

For this content the use case narrative is:

Title – Technology viewpoints
Narrative – (see below).
Primary Actor – student.
Scope – web server.
Level – Masters.
Stakeholders and Interests – independent study for students, may feed into tutor conferences.
Preconditions – none.
Minimal Guarantees – none.
Success Guarantees – Successful completion of two grids.
Main Success Scenario – grids submitted to portfolio.
Extensions – none.

The narrative element of the use case narrative expands to give a structured educational description:

Title – Technology viewpoints.
Provided by – The Open University.
Pedagogy/Type of learning – individualized linear.
Description/Context – Technology viewpoints looks at the differing views of technology focusing on technology and social determinism and utopian and dystopian views. Relevant readings around these topics are provided. A grid is formed from these two continuums and students are required to place their own views on the internet on this grid. In activity two they do some more reading and place the views of the authors on the grid.
Learning objectives – An understanding of different viewpoints relating to technology; Experience in appreciating the viewpoints of authors.
Roles: - only one main role and actor: student.
Different types of learning content used – the following web content is used:

- Introduction
- Reading list (links to external articles)
- Activity 1 description – place own views.
- Activity 2 description – place views of others.

Different types of learning services/facilities/tools used – Two dimensional grid plot; References chapter set book.
Different types of collaborative activities – none.
Learning activity workflow – There are four activities: Introduction; Reading; Own views; Other views.
Scenarios – the same content may be used for face-to-face and distance learning.
Other needs / Specific requirements – none.

A simple UML view of this structure is shown in figure x. In this case it is assumed that a linear flow results; if a more complex example was used, e.g. with different orders possible, such diagrams can be extended with “swim lanes” to represent these alternatives.

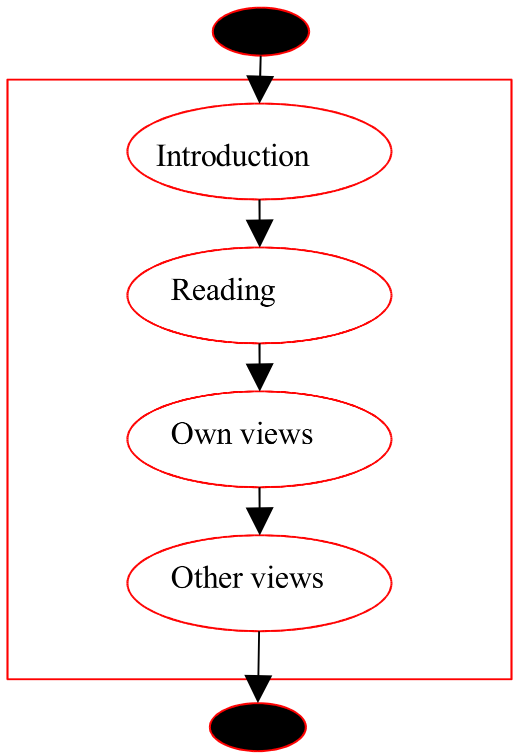

Figure 2: UML representation of the activity in the design

The last stage in producing a LD is to encode it into XML. This has been carried out by hand for this existing example, in the near future it is expected that editors will be available to allow this stage to be completed with more assistance.

The XML represents the LD and includes references to the resources and tools needed (the “environment” in LD terms) to provide a complete description of this unit of learning. Taking those files and passing them through the CopperCore validation produces a new set of XML files that instantiate the specific aspects needed, in this case the way to complete the grid, the set of readings and the timescale for the task to be completed. Rendering that output for the learner completes the LD Player and gives us a way to deliver the design to a learner. For us this is an important stage in developing use of LD, however it is even more important to reflect on what we have produced by following the separation encouraged by LD. We now have a description of the design that can be manipulated in several ways: we can change the presentation; we can change the tool sets; we can change the resources. Further than that we can also revise the initial design conditions – for example we can add in a collaborative sharing of views by making the grids common across all users and synchronising the activity in some way.

This opens the way for a structure to parameterise this sort of design. In this case the parameters could be:

- Whether collaboration is expected
- Resource sets (URLs) – for readings and for activity descriptions
- Specification for the output tools (grid creation)

From this we end up with a prototype LD wizard that is very simple but actually powerful to carry out variations on this design. A mock up interface is shown in figure 3.

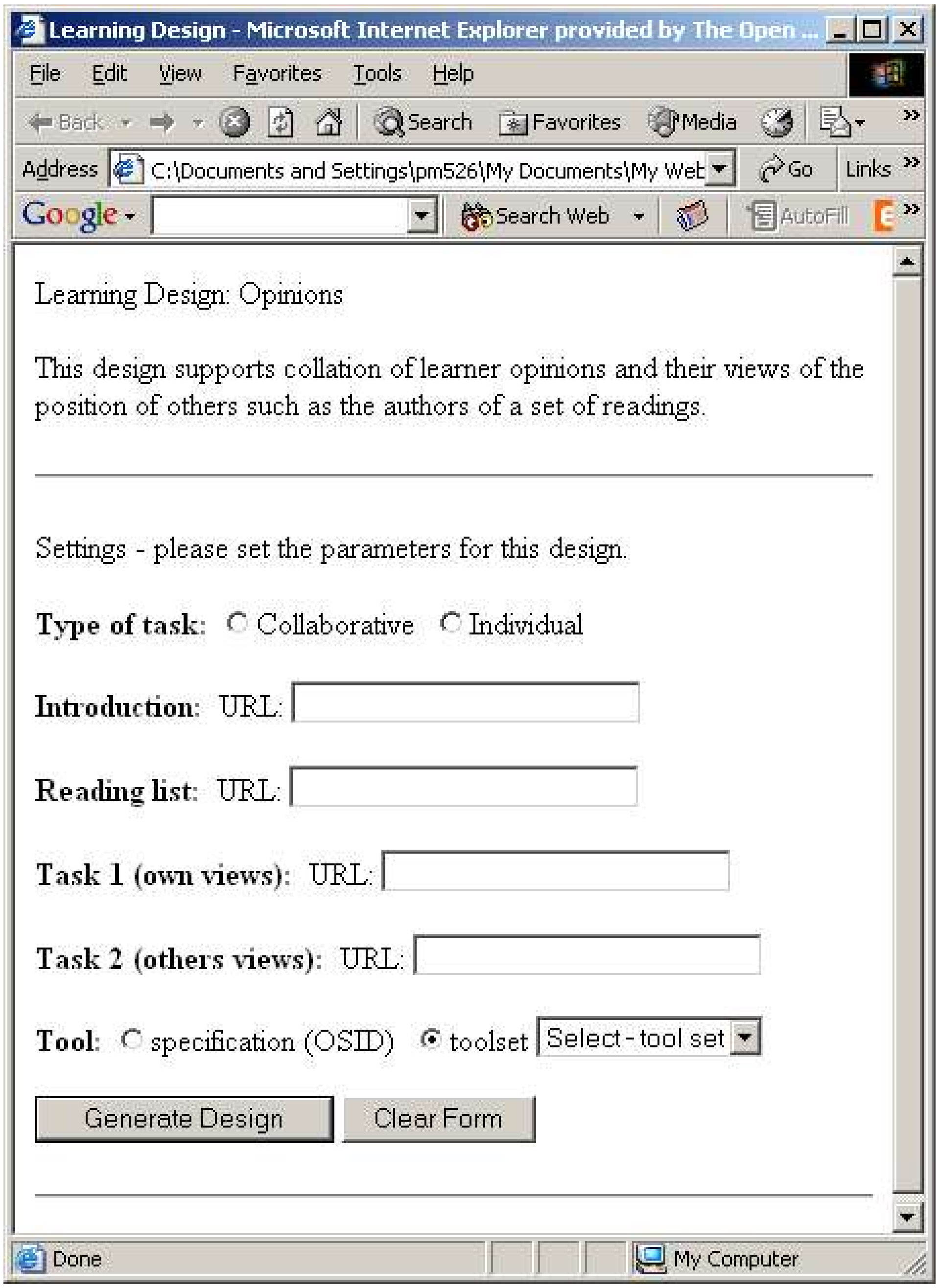

Figure 3. A mock-up of a Learning Design wizard which feeds into CopperCore.

This example has served to illustrate the aims for the joint work between the OUUK and OUNL. We also believe that a complete learning design system can offer scope for managing the sharing of the designs and analysis of their structures. In the next section we outline the development plans and architecture to take forward some of these ideas.

Learning Design Instantiation Development Process

In the original planning for the LD instantiation at the OUUK we expected to spend initial effort on the engine components and interpretation and validation of the LD. The release of CopperCore has enabled us to avoid this development stage and change our focus to the generation and management of design templates and the representation of service for playing. The development process will now be conducted in two stages.

Stage one is to develop a LD system which will run a single instance of a course based on level A LD. This stage has three development strands which weave together to form the system:

1. The first strand of development is being carried out by OUNL and involves the creation of the CopperCore system. CopperCore acts as the verification and validation mechanism for the IMS Learning Design. This also involves the extension of CopperCore to interact with external tools (e.g. authoring environments) and the Learning Design Player.
2. The second strand is the development of the Learning Design player, which will talk to CopperCore and will deliver the course material. The player will also be able to support service calls to and from OU systems or other external services. The player is being developed by the OUUK using Java using the Borland JBuilder X Enterprise environment.
3. The third strand will be the exploration of authoring tools (LAMS, Reload etc.) and the development of a bespoke authoring environment to support the input of OU course materials. A focus of this work is to produce a "LD Wizard" to simplify handling variations in designs and to encourage the reuse of standard specifications for services.

The first stage of the development process is expected to take around three months, after three months we will have expected to have produced a system which can deliver simple course objects (see figure 4 for details).

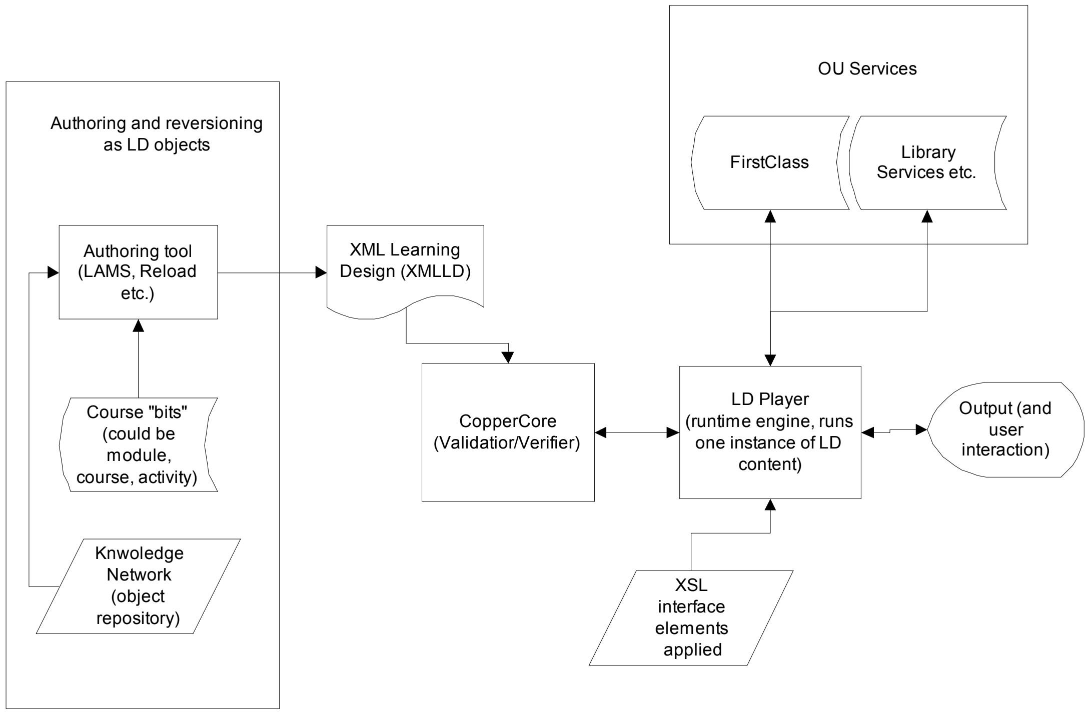

Figure 4: System design for the Learning Design Instantiation built on CopperCore

Stage 2 of the development process is to extend this architecture to deal with levels B and C of the IMS Learning Design specification. The second stage will take a further twelve to eighteen months and should produce a system which will accommodate all course delivery; this will involve the exploration of online course models and authoring systems. The planned approach is to use Simple Object Application Protocol (SOAP) to allow operation as web services. This approach will examine the work of the SAKAI project [www.sakaiproject.org/sakaiproject/] in the US to look at the applicability of their generic tool portability profiles (TPPs) which build on the Open Knowledge Initiative Open Service Interface Definitions (OKI OSIDs)[http://web.mit.edu/oki/specs/].

The development relies on the creation or adaptation of course materials supplied by OUUK. These course materials will be stored in a course object repository within the Knowledge Network [http://www.open.ac.uk/kn], a powerful system developed for knowledge sharing built using a SQL database with ColdFusion middleware.

Conclusion

The approach taken in this work is to be pragmatic; we recognise the opportunities for a new style of tool that is web-based, values designs and separates out presentation issues. Such an approach is ideally suited to a modern component/services view of the tools available to students and offers a way to break the monolithic and proprietary approach encouraged by VLEs. Work within the OUUK on VLEs has found that their main value comes when there can be commitment to a single architecture; however the scale and existing practice makes such a single approach difficult to achieve and likely to introduce too much inflexibility. A design based approach on the other hand allows the existing division between academic work and presentation work to be emphasised and assisted. Our early work on Learning Design shows that it can provide a way to help represent both specific and generic designs and provide part of the solution. We believe this can then be a catalyst for overall change that will encourage reuse and speed up adoption of new solutions as components available to the learners. The collaborative work under way will develop the tools to assist this and align them with theoretical approaches to help us judge their value. This paper presents initial results as exemplars from real courses and an architectural design; this is now being translated into a programme of work to provide the tools needed by ourselves and the wider community.

References

Koper, E.J.R., & Olivier, B. (in press/2004). Representing the learning design of units of learning. Educational Technology & Society.

Hummel, H. G. K., Manderveld, J. M., Tattersall, C., & Koper, E. J. R. (2004). Educational Modelling Language and Learning Design: new opportunities for instructional reusability and personalised learning. Int. J. Learning Technology, 1(1), 111-126.

Engeström, Y. (1987). Learning by Expanding: An Activity-Theoretical Approach to Developmental Research. Helsinki: Orienta-Konsultit Oy, Finland.

Weller, M. J., Pegler, C. A. and Mason, R. D. (2003). Putting the pieces together: What working with learning objects means for the educator, Paper presented at the Proceedings of Elearn International. Edinburgh, Feb 2003. Retrieved March 19, 2004 from http://www.elearninternational.co.uk/2003/docs/presentations/ref_papers/wel.zip

McAndrew, P., Brasher, A. and Hardy, P. (2004) Determining research questions in e-learning. To Appear in Proceeding Networked Learning 2004, April 5-7 2004, Lancaster, UK. Retreived March 19, 2004 from http://kn.open.ac.uk/public/document.cfm?documentid=4329

Laurillard, D & McAndrew, P. (2003). Reusable educational software: a basis for generic e-learning tasks, in Resources for Networked Learning ed. A. Littlejohn, Kogan-Page, UK.

IMS LD (2003). IMS Learning Design. Information Model, Best Practice and Implementation Guide, Binding document, Schemas. Retrieved March 24, 2004, from http://www.imsglobal.org/learningdesign/index.cfm

Dalziel, J. (2003). Implementing learning design: the learning activity management system (LAMS). ASCILITE. Retrieved January 22 2004. http://www.melcoe.mq.edu.au/documents/ASCILITE2003%20Dalziel%20Final.pdf