

Practical and Pedagogical Issues for Teacher Adoption of IMS Learning Design Standards in Moodle LMS

Anders Berggren¹, Daniel Burgos², Josep M. Fontana³, Don Hinkelman⁴, Vu Hung⁵, Anthony Hursh⁶, Ger Tielemans⁷

Moodle Community Learning Design Book Study Group

¹IKT-Pedagogen
E-learning
Consultancy
Pers väg 14
S-794 92 Orsa
Sweden
<http://www.ikt-pedagogen.se/moodle>

²Educational
Technology Expertise
Centre (OTEC)
UNFOLD Project
OUNL
Valkenburgerweg,
177 PO Box 2960
6419 DL Heerlen
The Netherlands
<http://www.ou.nl>

³Facultat de Traducció
i Interpretació
Universitat Pompeu
Fabra
La Rambla 30-32
08002-Barcelona
Spain
<http://www.upf.edu>

⁴University of Melbourne
Faculty of Arts, Horwood
Language Centre
Parkville, VIC 3010
Australia
<http://www.hlc.unimelb.edu.au/>

⁵ICT Center for Education
Ministry of Education and
Training
30/18 Ta Quang Buu - Hanoi
Vietnam
<http://hop.edu.net.vn>

⁶Department of Educational
Psychology
University of Illinois
505 E. Armory
Urbana-Champaign
Illinois 61820
United States of America
<http://moodle.ed.uiuc.edu>

⁷Het Stedelijk Lyceum
Twente University/ELAN
Mariëndaal 55
7544 NG Enschede
The Netherlands
<https://studiewijzerplus.nl>

Abstract

Integrating the specifications and tools for IMS-Learning Design (IMS, 2003) into Moodle (Moodle, 2003), an open-source Learning Management System (LMS), is not just a technological question, but also relates to practical, pedagogical, and philosophical issues. This study documents the discussions and experiments of a team of teachers active in the Moodle community who are concerned with the development of international standards in future versions of Moodle. In the course (Moodle, 2005a) of studying the book, Learning Design (Koper and Tattersall, 2005), participants analysed the implications of integrating the LD specification into Moodle and the operation of various LD tools (Coppercore, Reload) and related tools (LAMS) within the Moodle environment. These differences were then summarized into general implications for future versions of both Moodle and Learning Design. This study concludes that continued, open dialogue between teachers and developers of both LD and Moodle is necessary to achieve transparent integration.

Keywords: Moodle, IMS Learning Design, Integration, Specification, Learning Management System, Open Source

1. INTRODUCTION

The creation of any international standard is a complex endeavor, and most particularly in a practice-based craft such as teaching and learning. Educational stakeholders have an interest in a common method of exchange across borders, languages, codes, venues, methods, philosophies, and interfaces. Until now,

however, this exchange has been limited to printed materials, a costly and physically limiting media. Early attempts at digital standards have focused on narrow areas such as quiz question packaging or sequenced content. Yet, teachers in particular are hungry to share full courses and learning scenarios, complete with content and processes that they have found useful. The IMS-Learning Design specification (LD) is one attempt to bring that fuller picture to electronic exchange that can theoretically include all forms of highly complex and flexible learning for both online and face-to-face learning venues.

Traditionally, the design of pedagogy has been the realm of expert instructional designers, textbook authors, and software engineers. With the advent of easy-to-program web scripting languages and simplified digital authoring software, teachers are playing a greater role in the creation of learning materials and designs. Furthermore, the popularity of open source course management systems with pluggable modules and point-and-click configuration has allowed teachers to experience unprecedented freedom of design. Now these teachers want to share learning objects or learning units with each other, first in teams, then across departments, and now amongst any institution using any kind of system. That is the emergent demand which leads to their interest in international standards. As the role of teachers grows, we see other stakeholders such as engineers, academics, developers and IT professionals playing a comparatively less directive, but more supportive role in the co-creation of these standards. Educators active in the Moodle community are especially interested to join in this dialogue with the LD community. A final stakeholder, the learners, are not the main subject of this study, but we acknowledge their growing role and responsibility in the design of learning.

Moodle is an open source learning management system (LMS) that has maintained interest in the IMS-LD specifications over the past two years since community discussions began on this topic (Moodle, 2003; 2004a). At that time, it was noted that the Learning Design specification was the most congruent standard for Moodle, since it allowed for learning scenarios to be constructed as sequences of learning activities rather than being restricted to sequences of learning contents or objects. Although Moodle can be used for many kinds of educational applications, it is based on socio-constructivist principles (Dougiamas, 1998; 2000) and most suited for an educational approach involving interaction amongst people rather than transmission of content. Furthermore, the PHP scripting and modularity of Moodle even allows teachers to supervise the creation of new activity tools in the LMS—the emergence of the teacher-developer.

This paper represents a view not from Moodle programmer-developers, but from these teacher-developers who actively influence the application and pedagogical directions of Moodle. The overarching presumption we hold is that any learning design process must be intuitive and empowering for teachers, and not intended solely as the professional realm of instructional designers. Our primary aim is to discuss the pedagogical and philosophical aspects of the process of moving to an international specification, IMS-LD, and secondarily to illustrate that process with our initial testing of LD tools. Our research questions addressed in this paper are threefold:

- If IMS LD becomes the standard for the design and exchange of teaching tools and materials in Moodle, how will it affect teachers accustomed to the current design approach in Moodle?
- What are the attributes of the current Moodle way of design that we wish to preserve?
- What are some strategies for integrating LD into Moodle?

We do not reach definitive conclusions, but summarize our collective impressions in this paper. The sections of the paper include: 1) describing our method for investigation, 2) outlining some difficulties in understanding the relationship between LD and Moodle, 3) exploring LD and LD-related tools in a Moodle environment, 4) drawing implications for future Moodle versions, and 5) making similar implications for future development of the Learning Design specification and tools.

2. METHOD FOR STUDYING LEARNING DESIGN

In February of 2005, concurrent with the publication of the first book on Learning Design (Koper & Tattersall, 2005), the Moodle community began an online study (Moodle, 2005a) of this book using a

chapter-per-week format facilitated by members of the community with extensive experience using Moodle in formal and informal education. A separate course was set up on the moodle.org community site, and members were invited to join as either facilitators or participants. Ten members volunteered to serve as facilitators and another fourteen self-enrolled by contributing to the forum, “Why are you interested in Learning Design?”. The group represented a diverse background including secondary and tertiary level teachers from Australia, Vietnam, Spain, Netherlands, Venezuela, Japan, Germany, United States, Italy, New Zealand and Sweden. Occasionally, developers of Moodle and LD tools visited and offered comments. Eleven chapters were initially chosen based on relevance to pedagogical issues, rather than tool design. Facilitation of the discussions were based on the focus group protocol, a semi-structured methodology in qualitative inquiry (Krueger & Casey, 2000). Focus group discussion is useful when a research question into social phenomena cannot be clearly identified, requiring exploratory investigation to determine relevant issues (Morgan, 1996). The procedure followed in this case was for each facilitator to read the assigned chapter, then post 3-5 discussion questions for members to respond to. Over the course of the study, this process generated 21 forums, with over 200 topics and almost 2000 postings. In addition, each facilitator created a wiki (editable group document) to summarise their respective section. Finally, a moving wiki (repositioned week-by-week) was used to record notes by any course member on the implications of each section relative to Moodle and LD development. These notes and postings formed the basis for this summary document and the resulting commentary that has been subsequently incorporated. Collaborative writing by the seven person writing team was conducted both privately on a closed “Teachers Forum” and publicly in writing forums separated section-by-section.

3. HANDLING LD CONCEPTS AND OPERATION IN THE MOODLE ENVIRONMENT

The initial immersion into Learning Design gave us an experience of confusion over terms, concepts and tools. Our group constantly mixed discussions amongst conceptual points, codified specifications and multiple tools which are in various stages of development. Teachers will need to grasp these differences before a meaningful discussion can take place. This section begins a clarification of terminology and the functions, pedagogical descriptiveness, and styles of design such as bricolage.

3.1. FUNCTIONS AND TERMINOLOGY

IMS LD is a notation, a proposed standard for modelling learning scenarios, while Moodle is an LMS, a complete package for managing, designing and running courses. Thus the two do not compare directly, yet each uses a language to describe the process of designing a learning activity. The differences in the terminology are subtle and the absence of some concepts in each other’s lexicon is a useful indicator that significant differences exist. For example, in the Moodle approach to design, the base structure is a “course”, while in LD the principal term is a run of a “Unit of Learning” (UOL). A Moodle course includes user management, enrolment, learner monitoring, activity modules (tools), resources (attached files and links), all visibly arranged on a single main course page. In LD, the Unit of Learning is a packaged set of activities, roles, content, while Moodle has no direct equivalent (see discussion on pedagogical descriptiveness). While an LD UOL could be a whole course, in general, it is assumed that a number of UOLs will be assembled to make a full course. The assembling package is called an LMS, and LD does not directly attempt to model that total environment. Associated with IMS LD are various firms and organisations which have developed tools called “editors” which create designs, and “players” which run them for students. In Moodle, those two roles are integrated in one environment. Other differences are illustrated in Table 1, a first attempt towards a dictionary that translates the differences used in the lexicons of IMS LD and Moodle.

Terminology in IMS LD	Terminology in Moodle
Run of a Unit of Learning	Course

Unit of Learning	Course export/import file (without runtime data)
Activity-structure of type selection	Topics in a course
Learning activity with one single environment with one tool (depends on the activity)	Activity Module, Activity
Conference of type 'announcement'	Announcement
Conference of type 'asynchronous'	Forum
Conference of type 'synchronous'	Chat
Learning Object of type 'tool'	Wiki
Learning Object of type 'test'	Assessment
Learning Object of type 'tool'	Glossary
Learning Object of type 'tool'	Journal
Learning Object of type 'test'	Quiz

Table 1. Differences in Terminology for IMS Learning Design and Moodle LMS

3.2. PEDAGOGICAL DESCRIPTIVENESS

The Learning Design specification excels at modelling the structured sequencing of activities/resources and the roles of learners and teachers. A unit of learning in LD is multi-dimensional (Olivier & Tattersall, 2005), including a collection of activities that can be forced-sequenced, conditionally-sequenced, or non-sequenced. Content can be embedded within the unit of learning, not just separated in a simple sequence. Currently the Moodle editor has no system creating forced paths of activities, just place holders for separated activities and resources inside a course, which are only visually “connected” in a vertical column of the main interface. Content is also separated as individual files and links, called “resources”. It is one-dimensional in the sense that each resource and activity module (“tool” in LD) is totally independent and arranged under topic-labels, not formal UOL structures. This is an advantage in terms of ease of design, but a disadvantage when a particular learning unit needs to be containerized and

component dependencies described. In setting up a Moodle course, there is a blank column of topics or weeks--almost no structure “out-of-the-box”, but an arbitrarily complex structure can evolve over time. Learners experience maximum control because they can visualise the whole structure and are given full access for free inspection, skipping, jumping back anywhere on the main course page. Teachers, as well, as they edit in Moodle, are often given a start with a set of preformatted choices, but with freedom to reconfigure. This could be called “open learning design”. However, in some LD editors, such as Reload, a teacher starts with an empty canvas and can decide to design anything, but without the initial prompts to spur/constrain creativity. In this mode of “fixed learning design”, an LD editor allows the learning designer to decide what parts of a learning flow control are “automated”, what parts shall follow hard coded sequencing rules (“conditions”, defined by the learning designer), and what parts are just containers for more or less freely negotiated social interactions.

In addition, roles in Moodle are limited to “teachers”, “students”, “course creators”, and “administrators”. Moodle tacitly assumes that the learner's role will remain the same throughout the course. While a learner can be switched to a teacher role in Moodle, only one role can be played at a time and reassignment requires manual intervention by a course instructor. In future implementation plans for Moodle roles (Moodle, 2004b), an unlimited number of definable roles can be created, allowing specific editing and access rights to a defined role. For example, a group leader role might be allowed to edit quizzes, open forums, or assess reports. In an LD “play”, actors assume roles and sub-roles around the generic types of “learner” and “staff”. Although the LD specification does not limit editing rights in roles, current LD tools do not seem to be able to grant editing rights to learner roles. In the LD specification, roles are more complex, with multiple roles and conditional roles possible. From a teaching perspective, the eventual aim in any learning design tool is to allow instructor/facilitators to assign virtually any non-administrative role to a learner. Learners will become tutors of other learners and need powers to assess, plan, and manage their groups. Pedagogically, many Moodle teachers strive to create a learning environment for students where they get choices (and the freedom to make mistakes). This requires tools that support students with self-monitoring tools (mirrors) covering processes like self-planning, time-management, reflection, re-planning, choice in difficulty level of the activities. LD must thus allow students to play the design role, giving them editing rights, not just playing rights. A consequence of this pedagogy is that teachers play less of a design role, and more of a facilitator or coach role. It is a complex and heterogeneous process. Complex arrangements cannot be designed without describing and specifying the details and combination of the details of the coach role and the self-coach role.

Finally, the composition of groups within Moodle and LD are evolving. Student-centred, project-based, and socio-collaborative learning practices place greater emphasis on group-based configurations of learners (Jonasson & Land, 2000). The act of group formation may include self-organised, teacher-assigned, or automated assignment according to project interests. Multiple, simultaneous groupings are a necessary requirement as each learning unit has its own collection of groups, each of which may overlap in time. Moodle’s group function is for a course-wide, single configuration, useful for defining cohorts that do not change during the term of the course. In the LD specifications, the group functionality is based on “role-concept”. Some LD tool designers, such as LAMS and elive LD Suite, found this approach to be less intuitive and extended the current LD specification on groups. This extension is an open question for future discussion.

3.3. BRICOLAGE

One of the most striking features of the design approach favored by Moodle is the ease with which course materials can be developed and refined in an iterative fashion. This strategy of course development is very much in keeping with the notion of bricolage (Papert, 1980; Turkle & Papert, 1992), and what was earlier called “open learning design”. By contrast, the current implementation of Reload with CopperCore distributes a Unit of Learning in a fixed form, unalterable while instruction is in process. A fixed learning design process is useful in some situations, in other situations it may be difficult to adapt the UOL to handle unforeseen circumstances (either emergencies or unanticipated pedagogic opportunities), particularly when they occur after instruction has begun. For example, Ching, Hursh, & Scagnoli (forthcoming) discuss their discovery that the students in an introductory educational technology class had a strong interest in weblogs, and the ease with which Moodle allowed them to adapt the in-progress course to place more emphasis on weblogs. Similarly, McAndrew & Weller (2005, p. 288) refer to issues with the

“implied prescriptive nature” of the LD design approach because it seems to conflict with the “flexible and dynamic nature of e-learning”. In addition, many instructors who use an LMS to accompany their face-to-face class build their course week-by-week, redesigning the plan both during and after the class. These kinds of “bricoleur-teachers” would prefer to continually capture their learning design after-the-fact and then share these evolving units in a wider community of teachers. Another example of bricolage is this research study, organised as a Moodle course, which began devoid of any design. Taking the LD book as a “resource” or focus point, we organised discourse around chapters. The content (user postings) and choice of tools were added and rearranged week-by-week (during run-time). This is more than just “filling” a forum during runtime.

The architecture approach of Moodle and LD are quite different. Moodle is based on creation and modification on-the-fly and LD is based on splitting out design-time from run-time, like all of the e-learning specifications. Moodle allows technically-naïve instructors to create useful learning scenarios almost immediately, and then progressively refine them as their skills improve. This may be a critical factor in Moodle's popularity with teachers. The results of our group's ability to operate LD tools was mixed. One group member found LD tools such as the combination of Reload with CopperCore require much more front-loading of skills before useful results can be achieved, and iterative development was inconvenient at best. However, another group member found he was able to produce units using the same tool with 30 minutes of instruction. ASK LDT is another editor that may provide an easy-to-build approach that is focused more on the author's perspective, than the raw specification.

This is an important strategic question for Moodle/LD integration. If it is likely that the average teacher will be uncomfortable leaving the familiar Moodle environment to author a unit of learning with a separate LD tool, then several other strategies must be considered. On-the-fly creation within Moodle, with a subsequent generation of the LD specified format via an internal Moodle process seems like a more appealing option. This could be a “template editor” that would support the creation of more course formats that support roles and conditions. Thus for a Moodle course/UOL to be LD compliant, a way of “capturing” an end state (and stripping the user data) will need to be developed. An improved XML export system for Moodle that supports LD functionality and specifications may prove to be a not so difficult way to maintain the bricolage design approach.

4. TESTING LD TOOLS WITH MOODLE

There are several IMS LD related tools currently available: a) engines, b) editors and c) players. In addition, there are more tools in development which combine editing/playing and add GUI-based interfaces. In the following section we comment on how these different kinds of tools could be integrated or used in conjunction with Moodle.

4.1. LD ENGINES

CopperCore (Vogten and Martens, 2004) is an engine which implements all the levels (A, B, C) of the IMS-Learning Design specification. CopperCore is currently the only LD engine available and has been extensively tested with a set of examples on Levels A and B (LN4LD, 2005) and conforms perfectly to the LD specification. CopperCore provides three APIs: CourseManager, which provides administrative functions (users, runs, roles, publications); LDEngine, which provides run-time behaviour (activity trees, environment trees, content, completions) and Timer, which provides time-triggered events (various timed completions). CopperCore also provides a library for validating routines and the LD manifest. It is important to emphasize that CopperCore is an engine, rather than a learning environment or a management system, so it does not provide any user interface for creating LD packages. In other words, CopperCore was devised to run IMS LD packages not to design them or edit them.

4.2. LD EDITORS

LD editors are tools which create UOLs according to the LD specification. Table 2 provides a list of five examples of LD editors.

Nr	Tool Name	Link	Author	Levels
1	CopperAuthor	www.copperauthor.org	OUNL	A
2	Reload LD Editor	www.reload.ac.uk/ldeditor.html	Reload	A,B,C
3	ASK LDT	www.ask.iti.gr	University of Piraeus	A,B
4	Mot+	www.liceftelug.quebec.ca/gp/eng/productions/mot.htm	University of Quebec	A
5	Cosmos	www.unfold-project.net:8085/UNFOLD/general_resources_folder/cosmos_tool.zip	University of Duisburg	A,B

Table 2. Examples of LD-Compliant Editors (May 2005)

Moodle's XML-based backup files can be opened by CopperAuthor and the Reload Editor by modifying the namespace reference, but this is of marginal use, since there is no correspondence between LD's XML schema and that of Moodle. All these tools are Beta releases and hence they are in a fairly preliminary stage in their development. It is expected that they will reach a higher degree of compliance with IMS LD in future versions. These tools could easily work alongside with Moodle as external editors of IMS LD files. With the other applications we have tested, namely ASK LDT, Mot+ and Cosmos, we could not find any way to open Moodle XML-based backup files. Of course, all of these applications can be also used as external editors.

4.3. LD PLAYERS

There are several LD players available: CopperCore Player (as a built-in component of the CopperCore engine), Reload LD Player (Bolton, 2005), SLED, and Edubox (Tattersall, Vogten & Hermans, 2005). The CopperCore Player is a working prototype to demonstrate how UOLs run, to check internal functionalities, and to publish instances, roles and users to the engine. It doesn't run outside of its engine and its interface is not very user-friendly. The second player, Reload, has just been updated and offers better support than the previous version for several elements and learning structures of IMS LD. It still does not implement all the LD Levels and features but its developers are continuing to work on it and they are confident they will achieve full conformance very soon. As was the case with the CopperCore player, Reload can be used with Moodle as an external web player. SLED is developed under the JISC eLearning Framework. It has delivered an open source player version that integrates services and further development is continuing at the moment. The Edubox player is a full featured EML and LD player that is used at the OUNL as part of their infrastructure. It can import/export LD through Educreator but it is not usable for small scale deployment because it can only run on large Unix machines currently.

One of the main goals of this section was to report on a test run involving Moodle and the CopperCore Player in which we tried to determine whether it was possible to integrate both systems in a simple

manner. After several attempts, we conclude that Moodle resources can be used inside the CopperCore Player simply by hyperlinking to them. The inverse situation, running the CopperCore Player inside Moodle, is not as convenient. The run of a CopperCore unit of learning can be linked as a “Web Link” resource (which causes the browser to navigate away from the Moodle site) but not as a “Web Page” resource (which can embed the resource inside the Moodle user interface). It may be that this problem could be avoided with minor changes in Moodle's HTML code. Further testing is needed to determine the extent of interoperability.

In general, we are considering at least two levels of integration, a first level with just links between the applications and a second level with two-way communication between the applications. The first level is quite easy to achieve with the current players and engine: a player can be linked from Moodle as an external resource appearing in a new pop-up window or even within the same window. They would be in essence two different applications communicating with each other. Alternatively, the player could be embedded in the Moodle core and executed as an additional module.

In the first scenario, specific linking arrangements have to be made depending on the player but in principle they would not involve any major technical difficulty. The second scenario, that of embedded integration, is considerably more difficult to achieve because of the current state in the development of the players and engine as well as the current stage in the evolution of Moodle. Therefore, the strategy for full integration is still rather uncertain. More extensive testing is needed to find viable ways to allow Moodle to communicate with the existing LD engine and players. The goal for integration is to allow data to be exchanged between different tools that are running at the same time. CopperCore has similar data-exchange issues with other applications such as QTI and SCORM engines. Finally, we note that IMS LD editors cannot be used to edit Moodle courses but, of course, this was not expected to happen until Moodle courses can be exported as IMS LD Units of Learning.

4.4. LD-RELATED GUI-BASED EDITOR/PLAYERS

Two drag-and-drop GUI-based editor environments were discussed in this study: LAMS and elive LD Suite. Elive LD Suite (2005) is not yet available for testing but was described as offering an intuitive GUI-based sequence-editing environment. LAMS has been publicly released and was examined for this report. LAMS (the Learning Activity Management System) is a software system based on the concept of LD theory which has been in use with teachers and students since mid 2003 (Dalziel, 2003). It is an LD-inspired tool for designing, managing, and delivering online collaborative learning activities. It is important to note that the creators of LAMS do not see this platform as a competing learning management system, but rather as an activity/UOL authoring tool that could be used in conjunction with many LMS. LAMS has an intuitive interface with a visual authoring environment that allows users to create sequences of learning activities with very little effort (LAMS International, 2004). Although it is not LD compliant, LAMS is based on LD principles and it intends to be LD Level A compliant by July 2005. The LAMS team has pointed out some problems with IMS LD that made it difficult for them to implement an intuitive system under specifications (Dalziel, 2005). Table 3 shows a summary of the capabilities of all tools mentioned in this section.

Package Name	LD editor	Non-LD editor	Drag/drop editor	LD Player	Administration
CopperCore Player	X	X	X	O	O
CopperAuthor	O	X	X	X	X

Reload	O	X	X	O	X
ASK LDT	O	X	X	X	X
MOT+	O	X	X	X	X
Cosmos	O	X	X	X	X
LAMS	X	O	O	X	X
Moodle	X	O	X	X	O

Table 3. Roles and Capabilities of LD and LD-related Tools

At the moment, LAMS is one of the most immediately useful tools for the Moodle community because of its ease-of-use and the willingness of the developers to adapt it into the Moodle environment. According to recently published development roadmap projections, Moodle (2005b) intends to integrate LAMS as either a new course format, a new activity module or both in version 1.6 as a step towards eventual LD compliancy in version 1.7. In a recent demonstration (Malikoff et al, 2005), the LAMS tool could be used within Moodle as an activity or a course format and a LAMS sequence could link to Moodle activities as resources.

Activities/tools in LAMS are similar in function to Moodle activity modules. Moodle 1.5 activity modules include forum, chat, survey, choice, assignment (including journal), resources, grouping, glossary, lesson, wiki, messaging, and optional modules such as book, database, and questionnaire. LAMS 1.0.1's activities are similar, including forum, chat, journal, survey, voting, submit files, share resources, grouping, resource and forum, Q&A+Journal, Voting+Journal, Chat&Subscribe, and Chat&Subscribe+Journal. Moodle has many other activities in development (for example: blog, database, project, document management). The number of activity modules in Moodle is greater than in LAMS, but both sets are capable of building a rich collaborative learning environment. The main difference is that a LAMS activity was built to be "Learning Design aware", while a Moodle activity is not. With LAMS, you can create a sequence of activities and set the order of activities. Then the created sequence is saved in a private or public repository. If an author needs to modify some aspects, it can be reloaded from the repository and changed. In addition, there is a special kind of activity in LAMS called a Parallel Activity (Ghiglione & Takayama, 2005) which allows a single person to conduct two streams of activities concurrently on a single screen. We list such activities here: Resource and Forum, Q&A + Journal, Voting + Journal, Chat & Scribe, Chat & Scribe + Journal.

The case of embedded resources was the easiest for Moodle/LAMS interaction. We put LAMS inside Moodle as a resource (a link to a URL). Of course, LAMS and Moodle must have the same session so that we have no login problem. Interoperability interaction was more difficult because Moodle 1.x was designed with no "Learning Design" framework in mind. Therefore, it is hard for Moodle to interact with any UOL. At the moment, LAMS exports/imports a sequence of learning activities under its own format. Obviously, LAMS cannot use the course data of Moodle and Moodle cannot understand a sequence of LAMS. This, of course, is the reason interoperable specifications such as IMS LD are needed. Finally, in the case of activities interaction, we found that activities of Moodle and activities of LAMS cannot exchange data or re-use one another because they do not have a common interface for interaction.

In its next version, LAMS 1.1 will have a common interface to interact with the LAMS core so that third parties can write new tools for LAMS 1.1. Moodle tools (activity modules) will probably be able to be used in LAMS 1.1 (with some small extensions). Finally, the Tools Integration Project (TIP) of JISC has been focusing on integrating some popular open source software (Bodington, LAMS, AMSTOIA) using a WebAuth single sign-on mechanism (Noble, 2005). In this project, LAMS will be able to interact with other systems easily. Moodle should consider adopting this kind of capability.

While these options of *external* tools may be useful in the short run for integration of LD into Moodle, a second question is how strategically the Moodle code could integrate LD *internally*. Similar to the SCORM integration in Moodle, we could do the following:

- a. Create an Export filter that is LD compatible. This allows the transport of Moodle courses to other LD compatible players.
- b. Create an Import filter that can read the Moodle LD application profile (so files that are exported with Moodle or created with external tools are compliant with the Moodle application profile)
- c. Include an LD viewer (similar to the SCORM viewer) that can view any imported LD file that is not conforming to the Moodle application profile.

These are questions that programmers and engineers will need to resolve. Yet from a teacher's point of view, as stated earlier, it would be far preferable to achieve this internal integration, to provide a seamless working environment for a teacher.

5. IMPLICATIONS FOR FUTURE VERSIONS OF MOODLE

Moodle has begun an ambitious effort to integrate Learning Design standards into its future versions. Currently, version 1.5 is not compatible with IMS-LD specifications. However, according to a May 2005 future roadmap projection (Moodle, 2005b), Version 1.6 will move toward integration with LAMS as either an activity or a course format. Version 1.7 will aim for "preliminary support for IMS-LD Level A, allowing import and export" and integration with some repositories. Finally, version 2.0 is intended to provide complete support for the IMS-LD standard, conditional activities, and groups/roles customization at site, course, and activity level. Along with this vision, there are several further implications for Moodle on this pathway: 1) bricoleur tooling, 2) UOL-style authoring, 3) XML code output, 4) roles/conditions/paths, and 5) goals for LD levels.

5.1. MAINTAIN BRICOLEUR TOOLING AND ADD AN INTERNAL UOL EDITOR

The first implication is for Moodle to maintain its bricoleur mode of design and operation as it incorporates the LD specification. The French word bricoleur is normally translated as "handyman" or "tinkerer". The pedagogic sense of the word was introduced by Turkle and Papert (1992) which grew out of an earlier use by Levi-Strauss (1962). The idea here is that there are two fundamentally different ways of approaching a problem. The "engineer" way involves making careful plans and writing everything down in full detail ahead of time. The "bricoleur" way is more of an organic process of iterative design and refinement. While each approach is useful, the advantage of software designed with bricolage in mind is that the users can start producing useful results immediately. If the software requires lengthy training before worthwhile results can be produced, most teachers will not use it unless forced to do so. Moodle is an excellent example of software designed for bricolage. A naive (or even technophobic) instructor can start doing useful things in Moodle with five minutes of instruction. Seeing an immediate positive result is a powerful motivating factor. There seems however to be no fundamental reason why LD could not support bricolage by altering the LD XML tree while the code was running, similar to the way you can use DHTML to alter web pages that have already been loaded (a procedure that tools like CopperCore can support). Consequently, if it is technically possible, we would favor the development of LD tools that support this work style (preferably internal to Moodle so that an environment familiar to users can be preserved).

5.2. CREATE UOLS FROM STRUCTURED SETS OF MOODLE RESOURCES, ACTIVITIES AND SERVICES

The Moodle interface is presently organised like a stack of “cards” laid out vertically down the screen. Each card is a square box that represents a week or a topic. A card typically contains a title and some activities and/or resources. Even though a Moodle card is an almost self-contained “piece of learning” and can represent rather complex learning scenarios, it is organized as a rather simple flat structure. The title, activities and resources simply appear one after the other without any other kind of link or internal connection that could provide additional structure or relationships among the different elements in the card. This structure is the most fundamental difference between the central elements in an LD-specified UOL, and a Moodle “card”. In a UOL, all of its parts are formally related to one another. A UOL typically involves resources and/or services sequenced or linked to each other in some way. In contrast to the flat structure of the Moodle cards, where all activities and resources are visible in the same way for all users, UOLs often involve layers deep of non-visible activities and resources that can be also sequenced or visualized in different ways according to the roles assigned to the different users. In Moodle, as we said, the unit is flat, with no hidden activities behind a title. The title itself is just a label. It cannot hide or pull along any associated parts with it by dragging and dropping.

We propose Moodle add an optional, richer structure to its cards or to the elements within its cards. In other words, incorporate the richer structure of an LD UOL within a Moodle course but also allowing the option of unstructured elements or components contained in a course. Likewise, it should also be possible to export an entire Moodle course as a UOL. Under this perspective, UOLs would become an additional type of building block in Moodle, next to the traditional flat cards, which the teacher or course designer would have available to construct a wide variety of learning scenarios. The complexity of this kind of design, however, would require a new authoring interface, such as the drag and drop tool developed by LAMS. These movable, swappable cards/units would then be the core objects exchanged in a Moodle repository that is LD-compliant.

5.3. GENERATE XML CODE FROM MOODLE DESIGNS AFTER-THE-FACT

Moodle needs an 'after-the-fact' tool that builds an XML model after a teacher designs and implements a course. This would 'capture' a model/scenario after or while the learning has taken place. In other words, we would imagine that as a course progresses, the LD tool analyzes the online patterns and produces an XML model. In addition, a manual editor could then add the face-to-face aspects to the model. Currently in Moodle, there is a basic process happening like this already. Behind the mask of the zip-backup is a non-documented XML-tree. Moodle will need to rework this tree in areas such as automated updating of resources to become fully compatible with LD. Moodle tends more toward what the authors characterize as 'server-centered' rather than 'manifest-centered', though there are some aspects of Moodle that are reminiscent of a manifest-based approach, in particular the XML format used for backups. It may be that the backup format could migrate toward a more LD-friendly structure without too much difficulty (perhaps through an XSL transformation). This, however, is a fixed state of a course at one point in time. That is useful for exchange, but of course does not show the changing patterns of a learning design over time, a picture that eventually will prove valuable.

5.4 ADD MULTIPLE, DEFINABLE, CONDITIONAL ROLES

Moodle needs to implement definable roles as outlined in the Implementation Plan for Roles (Moodle, 2004b), and should move towards the capability to incorporate multiple roles, conditional roles and temporary roles. One goal is to create an intermediary role between teacher and student—such as “tutor” with limited teaching permissions. We can also define roles such as mentor and mentee. These roles would then be defined at the site level, course level, and activity level, possibly allowing multiple roles within the same course. However, it appears that the LD concept can go further with “multiple” roles. We assume this means that someone could have several simultaneous roles in a course. For example, within the project area B, John is tutor, but in project area C, he is a novice student. This would be very appealing to instructors who would very much like to take a single activity and assign all of our students to be “teachers” in that activity alone. For example, teachers often ask students to create quizzes on paper, and then assign one to a teacher role to input the questions into Moodle. However, this would work more

smoothly if it could become a configuration option inside Moodle. Another concept is conditional roles. A student would automatically be given a different role when certain conditions are triggered. This operation is much like moving up to the next level in a game. To do this, the user tables may need extra fields to store temporary role flags (during a course) or even longitudinal flags (preferred learning style), and even the combination of these flags. That process could be easy, but the difficulty would be implementing the engine that evaluates a script against these roles.

5.5 AIM FOR LD LEVELS A, B, C

Currently, Level A export functionality is under development to be delivered in 2005. As Moodle doesn't allow yet the features of Levels B and C (like properties, conditions or notifications) it is not appropriate to suggest their direct implementation. Nevertheless, at least two points should be considered regarding LD levels. First, LD levels are a distinction for implementers, not users. They are levels of the effort to implement the related functionality, not levels of the complexity of the learning designs that are created with a tool. This can result in situations where one has rather simple learning scenarios (from a teachers point of view), but these cannot be implemented on Level A, because, for instance, certain properties are required. Second, when someone decides to start with a Level A implementation, this should be done with Level B and C "in mind". The implementation of a sequencing mechanism in terms of "acts", for instance, will vary considerably depending on whether we plan to extend it in the future with sequencing triggered by properties and conditions. There is also a limit as to how much complexity can be reduced when the views and needs of the different stakeholders are considered. For this reason, implementation of all the three levels should be our goal from the outset.

6. IMPLICATIONS FOR IMS LEARNING DESIGN: FUTURE VERSIONS

Creating the universal learning design protocol, Learning Design, like any evolving standard, is in a continual process of development. Teacher practitioners such as those in the Moodle community are eager to contribute to this development because of their enthusiasm to begin sharing materials and designs in an inter-LMS exchange system. In this section, we shall outline some implications for the further development of Learning Design from this teacher-developer perspective as LD moves to become more widely accepted as a language of exchange. We will separate our recommendations and implications on three levels, beginning with the concept or theory of LD, then move to the specifications of LD, and finally the implementation of LD tools. It is perhaps overly audacious on our part to suggest changes after only a few months of immersion, so we ask LD developers to accept our apologies for any incorrect assumptions or immature understandings as we try to grapple with the intentions and concretions of LD.

6.1. MAINTAIN CURRENT CONCEPTUAL FRAMEWORK

The conceptual framework of Learning Design is powerful and appears to hold the core requirements that users of Moodle value and require. It goes beyond single-learner-in-isolation standards, such as SCORM, to include collaborative modes of learning with flexible roles. The core principles or requirements are all in alignment with the principles that Moodle users would generally agree on. These eight principles defined by Koper (2005, p. 19) can be summarised as:

- LD must be comprehensive: including objects, services, activities, roles, solitary/group models.
- LD must support blended learning: face-to-face integration as well as pure online learning.
- LD must be flexible: supporting all theories of learning, pedagogically neutral.
- LD must describe conditions of learning: tailoring the design to specific learners or situations.
- LD must stimulate reuse: portability, arrange-ability, addition/subtraction of parts.
- LD must be standardised: operate with other standard notations (i.e.: IMS-QTI)

- LD must be automatised: provide a language for automatic processing
- LD must be abstracted: for repeated execution in different settings and people.

6.2. ALLOW PLURALISTIC DESIGN PHILOSOPHIES IN LD TOOLS

While these core requirements provide an excellent framework for exchange of learning, questions have been raised as to the design methodology of specific LD tools. In other words, while the LD specification aims to be pedagogically-neutral, the LD-tools may prescribe a particular design methodology. Implicit in the design of any learning activity is an epistemological question about the nature of design. The nature of design has been classically conceived in a “pre-engineer and run” paradigm. Diffusion models of innovation (Rogers, 2003) operate in a similar way. First, an innovator constructs a new design, and then the design is disseminated. In contrast to this, there is a translation/transformation model of innovation in which designs are co-created by environments and actors in a way that continually transforms the network of actions (Law, 2004). The properties of the design itself are actually less important than the reconfigured network of actions and the very process by which this network of actions and relationships is reconfigured in a learning community. This community-based, ecological paradigm of learning may be a theoretical concern that LD will need to wrestle with. The most common concern that facilitators in the Moodle community expressed was the pre-engineering mode of operation that they felt they were being forced into when working with Learning Design. Moodle itself offers three pre-engineered formats (topic format, social format, and weekly format, yet within the topic and weekly format it not necessary to pre-design any aspect of the course. In addition, formats in Moodle are pluggable, with new formats in development such the Project Format and the Sequenced-Activity Format (LAMS).

The design-on-the-fly ability of the Moodle LMS was a critical attribute that no one was willing to part with. One commenter said, "freedom *from* design is just as important as freedom *in* design". In other words, it might be productive to distinguish between different types of 'design'--a conscious/explicit process of design and an unconscious/non-explicit mode of designing and compare LD tools through that criteria. The ability to design unconsciously is an inherent and useful practice that is embedded in the daily routine of teaching. In some ways, Moodle emulates this non-explicit design. The ability of LD tools to offer similar freedom may have to do with their design philosophy or current stage of development.

6.3. EXTEND LD SPECIFICATIONS IN SERVICES FOR COLLABORATIVE LEARNING

Another question was raised about whether LD was sufficiently developed to handle all the social dimensions of learning in Moodle. For example, we noticed that two LD-related tools, LAMS and elive LD-Suite, had found it necessary to extend the specifications of LD to handle the complexity of groups in learning. These two tools use runtime extensions to manage group functions in order to work on collaborative learning. Besides, although it is not its final goal, LD could include some components or an extension of the specification to incorporate services, such as fora or online events, for instance. At the moment each tool implementer is free to chose their own implementation. This is only a standardisation issue when we want to exchange these specific implementations. IMS is not working on these additional specifications because some tool developers argue that exchange is not necessary. However, in other ways LD appeared to be very ambitious in some aspects of social learning. In Halm, Olivier, Farooq & Hoadley, (2005) we found the peer-to-peer model of collaborative learning to be beyond the boundaries of our current thought. If LD can accommodate that decentralised kind of learning, it should have little problem with the issues surrounding group organisation and operation. In addition, LD seems not to have a specific way to handle forums, but just makes a reference to them, perhaps so the LD package itself is not tied to any specific forum setup. Moodle allows a number of definable properties to forums, and the varieties of group process produced by these configurable rules can and should be modeled. In addition, Moodle has numerous ways of handling unstructured communication, not just for discourse (wiki, blogs, instant messenger), but also for structured data (glossary, blocks, database). New code may need to be written in LD players to make them operate smoothly with any forum-oriented LMS such as Moodle. In the JISC ELF framework, this is solved by using web services whenever a tool cannot connect with a run of a UOL.

6.4. CLARIFY ADMINISTRATIVE AND LEARNING SERVICES

The general purpose of LD is not to provide a set of services because LD is not a LMS. Moodle has a rich set of student-monitoring services such as Gradebook, Activity Reports, Block Reports, Logs, and Portfolios that are an essential part of the learning environment. Bearing in mind the current state of development of on-line learning environments, it is not an exaggeration to say that the usefulness of most UOLs will depend more and more on the appropriate integration and configuration of these types of components. While LD specifications are capable of modelling units of learning it would be needed to integrate both, administrative and learning services, in order to provide teachers and learning users of useful supporting tools. For example, Moodle's tight integration of activity modules and activity reports and the effect this has on teacher coaching of students demonstrates that many "administrative services" in the learning environment have an impact on the success of learning.

6.5. PROMOTE GUI-BASED LD TOOLS

The number of LD and LD related tools is growing rapidly. For Moodle users, LAMS is one of the most intuitive tools because teachers can create a sequence of learning activities by dragging and dropping. Most other tools (i.e. CopperAuthor, Reload) are designed for users who are familiar with IMS LD concepts (play, act, role-part, etc) and may be more suitable for developers and designers than for the average teacher. LD tools should be more intuitive and easy-to-use so that non-technicians can use them to create and exchange UOLs.

7. CONCLUSIONS

This paper has been a review by teacher-developers of the complexities of integrating the Learning Design concepts, specification, and tools with an open source LMS, Moodle. We have attempted to view the two from an outsider perspective, though as Moodle users, it is not always possible for us to avoid certain presumptions. The first section compared Moodle and Learning Design in its terminology and pedagogical descriptiveness, and the contrast between bricoleur and pre-engineered design. The following section examined current LD tools and found that the distance for integration was far closer than we imagined. The LAMS/Moodle integration was an encouraging step towards LD compliance.

In the fourth section, implications for Moodle were outlined. The Moodle community needs to consciously preserve its intuitive structure for designing courses. A post-run capturing of LD-based XML schema will be needed to achieve both LD compliance and bricoleur design. Multiple roles in a structured unit of learning, with conditions and paths, are future requirements. Levels A, B, and C of the LD specifications should be the goal with an eventual internal Moodle editor for creating LD UOL.

In the fifth section, our group suggested implications for the further development of the LD specification and tools. We found the current conceptual framework to be comprehensive and very appropriate for modelling education in Moodle. With respect to the specification, we expressed concern that bricoleur-style design philosophies, collaborative learning complexities, and comprehensive learning services be well accommodated. Finally, in LD tools, we would support concurrent development of LD repositories to provide support for the exchange of UOL. In addition, it is important to promote intuitive design environments that are teacher and learner-friendly. These implications are summarised in the Table 4.

Implications for Moodle	Implications for IMS LD
1. Maintain bricoleur tooling & add internal UOL editor	1. Maintain current conceptual framework

2. Create UOLs from structured sets of Moodle resources, activities and services	2. Allow pluralistic design philosophies in LD tools
3. Generate XML code from Moodle designs after-the-fact	3. Extend LD Specifications in collaborative learning
4. Add multiple, definable, conditional roles	4. Clarify administrative and learning services
5. Aim for LD Levels A, B, and C	5. Promote GUI-based LD Tools

Table 4. Implications for Moodle and IMS LD Integration

The process of integrating the LD specification into Moodle is happening at a far higher pace than anticipated when we began this study three months ago. Step-by-step integration initiatives are already underway. With a distributed, online learning community, teacher-developers from secondary and tertiary institutions were able to grasp much of the complex conceptual framework of Learning Design and dialogue on the issues. This framework is now undergoing a translation into practice and will continue to be transformed as teachers and learners take fuller ownership of IMS LD.

ACKNOWLEDGEMENTS

Our sincere thanks for the following contributors, facilitators, and editors that made the Learning Design Book Study (March–May 2005) such a success. Sorted by forename:

- Bernhard Zech (elive LD Suite Project)
- Chris Kew (UNFOLD and Reload Projects)
- Dan Fleming (LD Book Study Facilitator)
- Dean Shankle (LD Book Study Facilitator)
- Ernie Ghiglione (LAMS Project)
- Gabriela Díaz Antón (Academia Interactiva)
- James Phillips (LD Book Study Facilitator)
- Joyce Smith (Moodle community leader)
- Martin Dougiamas (Moodle Lead Developer)
- Rob Koper (Open University of the Netherlands)

REFERENCES

- Bolton University (2005) The Reload LD Player and LD Editor. Retrieved 22 May 2005 from <http://www.reload.ac.uk>
- Ching, C. C., Hursh, A., & Scagnoli, N. (forthcoming) "Users and Producers: Constructivist Experiences with Open-source Courseware for Post-graduate Teacher Education".
- Dalziel, J. (2005) From reusable e-learning content to reusable learning designs: Lessons from LAMS. Retrieved May 7, 2005 from <http://www.lamsfoundation.org/CD/html/resources/whitepapers/Dalziel.LAMS.doc>.
- Dalziel, J. (2003) Implementing Learning Design: The Learning Activity Management System (LAMS). Retrieved 7 May 2005 from <http://www.lamsfoundation.org/CD/html/resources/whitepapers/ASCILITE2003%20Dalzie%20Final.pdf>.
- Dougiamas, M. (1998) A journey into constructivism. Retrieved 14 May 2005 from <http://dougiamas.com/writing/constructivism.html>
- Dougiamas, M. (2000) Improving the effectiveness of tools for Internet based education
- Elive LD Suite (2005) elive Learning Design. Retrieved 30 May 2005 from http://www.elive-ld.com/content/index_eng.html
- Ghiglione E. and Takayama, Y. (2005) LAMS Foundation: IMS Learning Design. Retrieved May 7, 2005 from http://users.ox.ac.uk/~jiscpub/JISC_TIP/documents/LAMS%20-%20IMS%20LD.doc.
- Graham, P. (2003) Beating the averages. Retrieved 2 May 2005 from <http://paulgraham.com/avg.html>
- Halm, M., Olivier, B., Farooq, U., & Hoadley, C. (2005) Chapter 11. Collaboration in Learning Design Using Peer-to-Peer Technologies. In Koper & Tattersal (Eds.) Learning Design. Pp. 203-213. Berlin: Springer-Verlag.
- Hopcroft, J., Motwani, R., & Ullman, J. (2000) Introduction to Automata Theory, Languages, and Computation (2nd Edition). New York: Addison Wesley.
- IMS (2003) Digital Repositories Specification. Retrieved from <http://www.imslobal.org/digitalrepositories/index.html>
- Jonassen, D. and Land, S. (eds.) (2000) Theoretical foundations of learning environments. Mahweh, NJ: Lawrence Erlbaum.
- Karte: Schweiz (2005) Retrieved May 24, 2005 from <http://map.search.ch/>
- Koper, R. (2005) Chapter 1. An introduction to Learning Design. In Koper & Tattersal (Eds.) Learning Design. Pp. 3-20. Berlin: Springer-Verlag.
- Koper, R. & Tattersal, C. (2005) Preface In Koper, Rob & Tattersal, Colin (Eds.) Learning Design. pp. v-xxvii. Berlin: Springer-Verlag.
- Koper, R. & Tatersall, C. (eds.) (2005) Learning Design: A handbook on modelling and delivering networked education and training. Berlin: Springer-Verlag.
- Krueger, R. & Casey, M. (2000) Focus groups: A practical guide for applied research. Third edition. Thousand Oaks, CA: Sage.
- LAMS International (2004) LAMS Brochure. Retrieved 7 May 2005 from <http://www.lamsfoundation.org/CD/html/resources/summaries/LAMS.Brochure.pdf>.
- Law, J. (2004) After method: Mess in social science research. London: Routledge.
- Levi-Strauss, C. (1962) "The Savage Mind" Chicago, IL: University of Chicago Press.
- Malikof, F., Ghiglione, E., & Dalziel, J. (2005) LAMS and Moodle Integration Walkthrough. Retrieved 6 June 2005 from <http://lamsfoundation.org/integration/moodle>

- McAndrew, P. and Weller, M. (2005) Chapter 17. Applying Learning Design to Supported Open Learning. In Koper & Tattersall (Eds.) Learning Design. Pp. 281-290. Berlin: Springer-Verlag.
- Moodle (2003) IMS Learning Design. <http://moodle.org/mod/forum/discuss.php?d=3758>
- Moodle (2004a) SCORM content packages. <http://moodle.org/mod/forum/discuss.php?d=1926>
- Moodle (2004b) Implementation plan for roles. <http://moodle.com/development/plans/roles.html>
- Moodle (2005a) Learning Design book study. <http://moodle.org/course/view.php?id=44>
- Moodle (2005b) Future Versions of Moodle. <http://moodle.org/mod/resource/view.php?id=3853>
- Morgan, D. (1996) Focus groups as qualitative research. Thousand Oaks, CA: Sage.
- Noble, H. (2005) JISC: Tools Integration Project, retrieved May 7, 2005 from http://users.ox.ac.uk/~jiscpub/JISC_TIP/Index.htm.
- Olivier, B. and Tattersall, C. (2005) Chapter 2. The Learning Design specification. In Koper & Tattersall (Eds.) Learning Design. Pp. 21-40. Berlin: Springer-Verlag.
- Papert, S. (1980) Mindstorms: Children, computers, and powerful ideas. New York: Basic Books.
- Sloep, P., Hummel, H. & Manderveld, J. (2005) Chapter 8. Basic procedures for e-learning. In Koper & Tattersall (Eds.) Learning Design. Pp. 139-160. Berlin: Springer-Verlag.
- Tattersall, C., Vogten, H., & Hermans, H. (2005) Chapter 19. The Edubox Learning Design Player. In Koper & Tattersall (Eds.) Learning Design. Pp. 303-310. Berlin: Springer-Verlag.
- Towle, B. & Halm, M. (2005) Chapter 12. Designing adaptive learning environments with learning design. In Koper & Tattersall (Eds.) Learning Design. Pp. 215-226. Berlin: Springer-Verlag.
- Turkle, S. & Papert, S. (1992) Epistemological pluralism and the revaluation of the concrete. Retrieved 24 May 2005 from <http://www.papert.org/articles/EpistemologicalPluralism.html>
- Vogten, H., Tattersall, C., Koper, R., van Rosmalen, P., Brouns, F., van Bruggen, J., Sloep, P., & Martens, H. (2004) Implementing a Learning Design engine as a collection of finite state machines. Preprint available at: <http://hdl.handle.net/1820/96>
- Vogten, H., & Martens, H. (2004) *CopperCore*. Retrieved 14 March 2005 from <http://www.coppercore.org>.

