Delivering courses modelled using IMS Learning Design

Delivering courses modelled using IMS Learning Design[1]

Colin Tattersall, Hubert Vogten, Francis Brouns, Rob Koper,
Peter van Rosmalen, Peter Sloep, Jan van Bruggen
Educational Technology Expertise Centre, Open University of the Netherlands,
Valkenburgerweg 177, 6419 AT Heerlen, The Netherlands

PREPRINT

ABSTRACT

E-learning promises efficiency of education. This promise is based on the economics of multiple delivery, whereby initial production costs for an e-learning course are recouped by delivering the course to different groups of learners at different times. This can only be realised by distinguishing between abstract representations of courses and instances of these representations involving specific sets of learners. This article provides an analysis of the requirements for multiple deliveries of courses. It describes the design of an approach which meets these requirements in the domain of integrated e-learning systems, together with experiences resulting from implementation of the design. The article concludes with a discussion of the approach.

Keywords: E-learning, Course Delivery, IMS Learning Design, Educational Service Providers

Introduction

Faced with increasing demand for education together with fixed (or declining) resources, Educational Service Providers (ESPs) are increasingly turning to technologies which support learning using the web, or e-learning. The move to e-learning opens many doors, including opportunities for ESPs to explore new approaches to teaching, and for learners to vary time and pace of study. One of the most powerful drivers for e-learning is an economic one which follows the well established economics of the publishing world—courses can be created once and delivered many times. Although each delivery incurs costs, these are marginal and more than covered by the fees and subscriptions paid by the material's consumers. In this way, over time, high initial production costs are first recouped and subsequently exceeded by revenues, yielding course profit. Substantial initial costs can be justified by informed market forecasting and used to invest in high quality learning experiences which might otherwise be impossible to finance.

These opportunities have sparked a proliferation of commercial and open source course delivery systems, also known as (web based) course management systems, courseware delivery systems, online educational delivery applications and learning management systems (see (Brusilovsky & Miller, 2001) for an overview).

[1] Correspondence concerning the article should be addressed to Colin Tattersall, Educational Technology Expertise Centre, Open University of the Netherlands, Valkenburgerweg 177, 6419 AT Heerlen, The Netherlands, Email: colin.tattersall@ou.nl, Fax: +3145 576 2907

Although the ‘create once, deliver many times’ approach is one of the foundations of e-learning economics (Sloep, 2003; Molyneux, 2000), it is often overshadowed and overlooked in the wider debate on re-use in e-learning. There, the focus is squarely on re-use to support the creation of new courses and is dominated by discussions of learning object repositories, and methods and techniques to support the creation of new materials (Downes, 2001; Friesen, 2001).

Consideration must nevertheless be given to the concepts required for successful application of the ‘create once, deliver many times’ approach in the design and development of integrated e-learning systems (Koper, 2003). Without such attention, processes and systems for the delivery of courses can lead to a ‘create once, deliver once’ situation where each delivery is associated with its own unique variant production. The consequences of this situation are often not felt immediately, since copies of productions can be made instantly and at negligible cost. However, the seeds are sown for future course administration and management difficulties which ultimately undermine the original economic case.

This article provides an analysis of the requirements for reproducibility in e-learning, where reproducibility refers to repeated delivery of e-learning courses in different settings with different participants. Thereafter, the article describes the design for part of an integrated e-learning system which meets the requirements, followed by a description of a production level implementation of the design. It concludes with a discussion of the approach.

Requirements Analysis

We distinguish between course enrolment, when learners sign up to participate in e-learning courses, and course delivery, the process by which learners are engaged in learning processes supported by e-learning systems. The problem area addressed by this article is the combination of course creation and management, and course delivery policy. The latter concerns the manner in which those enrolled for a course have their education delivered, focusing on when delivery occurs and how (in terms of cohorts and sets of learners).

Building on developments from the distance and open learning world, e-learning has extended the spectrum of delivery policies from the traditional university class, where class size, start and finish dates are fixed by the provider, to fully individualised situations, in which group size, time and pace of study may be determined by learners. Many possibilities can be identified, including situations wherein:

a course is run once only (then discontinued), with a single set of learners.
a course is run for several sets of learners. The rationale behind the dividing of learners into sets is here a logistical one for the ESP. The availability of staff resources to act as tutors might be constrained by institutional policy that the staff-to-learner ratio must never rise above a certain advertised maximum. Alternatively, the division might reflect simple physical constraints, such as classroom size for blended learning courses where groups of more than 40 cannot be accommodated for face-to-face sessions. In contrast, it might reflect the targeting of different geographical areas or market segments (for example running the course in the winter months and marketing to those seasonal workers fully employed in the summer months, and vice versa).
a course is run for (possibly several sets of) learners and the learners are divided into groups. In contrast to the previous possibility, the rationale here is pedagogical, reflecting a choice to pursue, for example, a group-based learning approach in which learners are

divided into competing teams, or a Problem Based Learning approach (Nulden, 2001). Similarly the use of computer-supported collaborative learning (CSCL) technologies might be associated with working in small groups.
a course is run only when, but as soon as, there are enough learners enrolled on it. Here, the decision might be a pedagogical one (group learning) or might reflect economic reasoning, such as the need to have a minimum number of learners to break even.
a course is run for each individual learner as soon as the individual's enrolment has been finalised.

ESPs may wish to adopt different delivery policies, either to gain competitive advantage through flexibility of delivery, or to reflect the stage of development of the organisation (starting with limited flexibility but increasing as the organisation's logistical processes mature).

Without an appropriate approach to delivering e-learning providing adequate separation between courses and their delivery to learners, ESPs run the risk of being forced down a path of creating course variants each time a course is run. This results in a 'create once, deliver once' situation. In situations involving large numbers of learners—see (Daniel, 1998) for some extreme cases—this situation becomes un-maintainable.

While targeting flexibility of delivery, providers must also be aware of legal obligations on retaining information in cases of dispute with learners. Information on both the structure and content of a course, together with that concerning its time-of-delivery and cohort size must be preserved, and the obligation may exist long after learners' participation in courses has ended. Providers making modifications to courses on-the-fly without paying sufficient attention to version management run the risk of losing law suits filed by learners who dispute their failing of course examinations.

Notwithstanding the need for effective version management of courses, providers need to be able to make minor modifications (i.e. without legal significance) to materials being used in running courses. Such modifications include correcting spelling errors in course materials, improving the readability of materials following learner feedback, and the updating of links to time-dependent material used in courses such as company yearbooks and governmental surveys.

We identify four requirements to be met by ESP processes and systems in the area of reproducibility:

The same course must be able to be delivered to different sets of learners without resorting to duplication of course structure and contents.
Deliveries must be able to be handled in an efficient way, and where possible, partly or wholly automated. Meeting this requirement further reduces delivery costs thereby strengthening the 'create once deliver many times' case.
Effective version control must be applied to courses.
Minor updates to running courses must be possible without disrupting on-going learning processes.

E-learning practice has not always taken these requirements into account, as noted by Porter (2001):
"In many cases the instructor is given training in a particular online development, delivery and management tool … and then the instructor proceeds to craft a course for online delivery. The pedagogical structures embedded within the instructional delivery tool are

tweaked to suit the needs of the class, the content or the particular instructional problem. In most cases, the courses are hand tooled and kept current through the intervention by the instructor over time”.

This point is echoed by Abdallah et al (2002) who note that in a commercial course delivery system “each module, lesson, Web page has to be duplicated in each course if needed”. Similarly faculties’ resentment of the time required to load and reload course materials is noted as one of the factors which leads to reduction in faculty use of course management systems (Morgan, 2003).
Such practice contrasts sharply with the publishing world-inspired production systems required to realise the ‘create once, deliver many times’ promise.

Design

The above analysis points to the need to distinguish between a course in the abstract, and its deliveries to different sets of learners. The IMS Learning Design Specification (2003b) provides an appropriate context within which to view this distinction.

An overview of IMS Learning Design

The IMS Learning Design specification provides a notational system to describe a ‘unit of learning’, an abstract term used to refer to any delimited piece of education or training, such as a course, a module, a lesson, etc. The notation is capable of describing a wide variety of instructional models, or learning designs, such as Competency Based Learning and Problem Based Learning.

One of the requirements the IMS LD specification is designed to meet is that of reproducibility—“the specification must describe the learning design abstracted in such a way that repeated execution in different settings with different persons is possible”.

In a unit of learning, people act in different roles in the teaching-learning process. In these roles, they work toward certain outcomes by performing learning and/or support activities within an environment containing learning objects and services to be used during the performance of the activities. In this way, a unit of learning represents more than just a collection of ordered resources to learn. It includes a variety of prescribed activities (problem solving activities, search activities, discussion activities, peer assessment activities, etc), assessments, services and support facilities provided by teachers, trainers and other staff members. Each learning design (contained in a unit of learning) must include a learner role engaging in one or more activities. The staff role is optional and can be used to model the activities of teachers, tutors, mentors etc.

Units of learning are created as so-called content packages—bundles of all resources required in the learning process represented by the unit of learning, including the learning design, physical files of educational material and hyperlinks to resources. A separate IMS specification governs the structure of such packages (2003a).

Through its use of roles, IMS LD abstracts from the details of specific learning situations and provides an appropriate concept to describe a course in the abstract: the unit of learning. It is units of learning that are created once, then delivered many times.

Moving from an abstract course to specific deliveries

The 'creation versus delivery' distinction reflects that between design time and runtime. The formal description of a learning process which results (at design time) from the use of the IMS LD notational system is interpreted (at runtime) by an IMS LD-aware software component, or IMS LD Player, in the same the way HTML is interpreted by a browser. Taking into account the requirements identified in the previous section, it is clear that the design time concept (the unit of learning) cannot be the same concept interpreted and used by the runtime software component. Addressing the reproducibility problem requires the introduction of an additional, runtime concept.

The need to establish a specific runtime concept related to the abstract design time concept of a unit of learning can be informed by the world of object orientation. Although the link between e-learning systems and object orientation has been examined in other work (Permanand & Brooks C., 2003; Douglas, 2003; Virvou & Tsiriga, 2001), such work has tended to focus on the re-use of learning objects at design time, that is, in creating new units of learning. In the context of reproducibility, the focus is on the move from design time to runtime. This is the process of instantiation, whereby an object class, modelling an abstraction, is used as the basis from which to create specific object instances. Following this line, we view a unit of learning as describing a class of possible instances, and we use the term instantiation to describe the process of transforming an abstract unit of learning into deliveries for learners (this is the assignment process described in (Widya, Volman, Pokraev, De Diana, & Michiels, 2002). The specific instances of a unit of learning are referred to as runs, defined as the combination of a particular unit of learning with an assigned community of users. Each run is assigned to exactly one unit of learning, but a particular unit of learning may have zero or more runs assigned to it.

Additionally, we exploit the notion of a publication, which is introduced to allow pre-processing of the contents of a unit-of learning for a run. Publications are not strictly necessary to meet reproducibility requirements, since the processing can also be achieved on-the-fly, but have proven useful in several situations in practice. The first involves the selection of alterative resources in different languages, as is the case when a course is run for different sets of learners with different mother tongues. Here, the pedagogical approach remains identical for the two groups but the resources utilised in the learning design differ, including instructions to learners and staff, materials to be read and exercises to perform. The content package representing the unit of learning consists of a single learning design with multiple alternative resources. These alternatives are split out during pre-processing for linking to particular sets of learners and staff in a run (for example, a run with French contents and a run with English contents in the Canadian context). In a similar vein, alternative resources can be selected for different media, such as a course offered both in printed form and over the web, or for different (mobile) devices, perhaps with differing display sizes and capabilities. Finally, publications can be exploited to accommodate variety in the formatting and styling of units of learning for different sets of learners, meeting both accessibility and re-branding requirements on course content. A full examination of the utility of publications is outside the scope of this article, but we note recent interest in the need to support re-branding of e-learning material (Canadian Department of National Defence, 2003).

This resulting combination of concepts is illustrated by the UML class diagram shown in Figure 1:

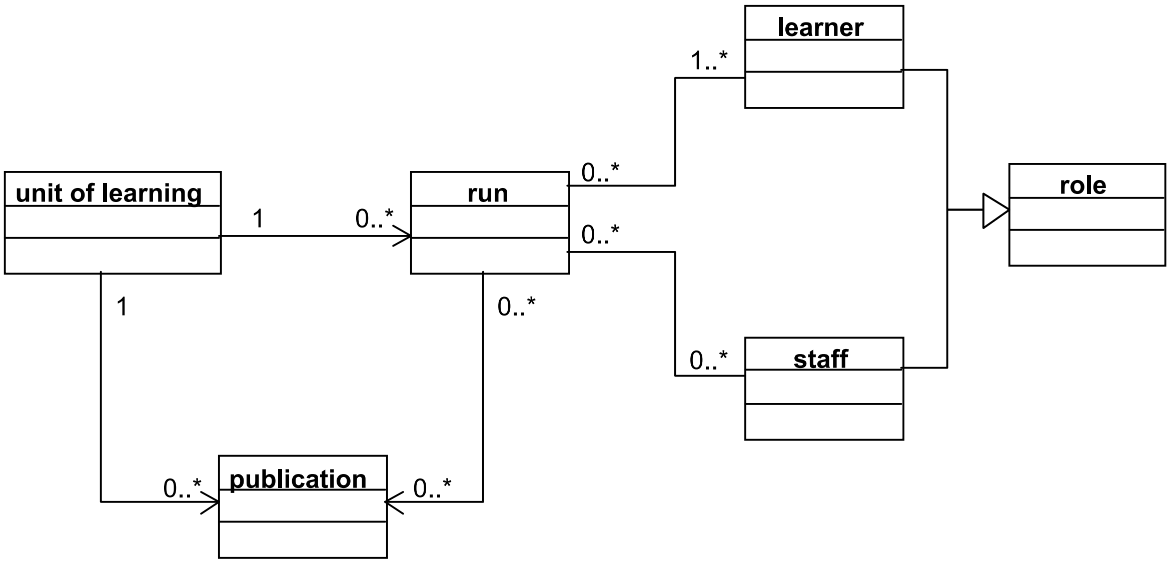

Figure 1. The relationships between units of learning, publications, runs, roles, staff and learners.

A run adds runtime information to a unit of learning by defining a start and end date and binding specific individuals into the roles modelled in the learning design part of the unit of learning.

The same unit of learning can have an unlimited number of runs. The notion of an abstract class (unit of learning) from which specific instances (runs) can be spawned is at the heart of the solution to reproducibility problems. Various delivery policies can be realised by creating multiple runs from a single unit of learning. In all cases, the 'parent' unit of learning is frozen and archived for future reference, with each 'child' run maintaining a link to its parent. The unique identification of a unit of learning using a Uniform Resource Identifier which is mandated by the IMS LD specification is also applied to each run.

Constraints on run creation

Mechanisms are provided in the IMS LD modelling language to help designers (at design time) indicate constraints on the creation of runs (at runtime). The mechanisms provide the basis for automation of run creation and build on the two general roles inherent in the specification—learner and staff.

Two of the constraint mechanisms are the "min-persons" and "max-persons" attributes associated with a role. The former specifies the minimum number of persons which must be bound to the role before starting a run and the latter specifies the maximum. Runs are generated using the constraints until the enrolled population of learners is exhausted. Note that if the attribute is not used, no restrictions apply to the number of individuals who can fill a role. This can be useful in situations where the number of individuals participating in a unit of learning is unimportant, such as is the case with fully individualised, self-taught courses.

By combining these attributes with the notion of a default run, the delivery process is opened to partial or full automation. If only one run is created for a unit of learning and it is designated as the default run, learners can be automatically assigned to participate in runs according to any min-persons and max-persons constraints. Therefore, we extend the definition of a run to include an attribute indicating whether or not it is the default run. Only one run for a learning design may be a default run.

To illustrate the utility of default runs, consider a cohort of 200 learners for a given unit of learning which has constraints indicating a minimum of ten and maximum of twenty individuals in the role of learner. As soon as the run is made available, ten runs are spawned automatically, each with twenty learners. Alternatives to full automation are also possible, whereby humans-in-the-loop are used to couple learners to runs.

Finally, a run progresses through a lifecycle, mirrored by its changing status—either waiting, active, stopped or achieved. When a run is first created it has the status 'waiting' meaning that users have still to be assigned to the run from the pool of enrolled learners before delivery starts. Delivery starts when the run status changes to active. As soon as all users have finished, the run gets the status of stopped, meaning that users can still access the learning design and the corresponding content contained in the unit of learning but no more interactions will be allowed. Finally, a decision can be made to archive the run, meaning that it is no longer available to the learners and staff, but all information is stored in an archive for future reference.

The final design is reflected in the UML domain model shown in figure 2:

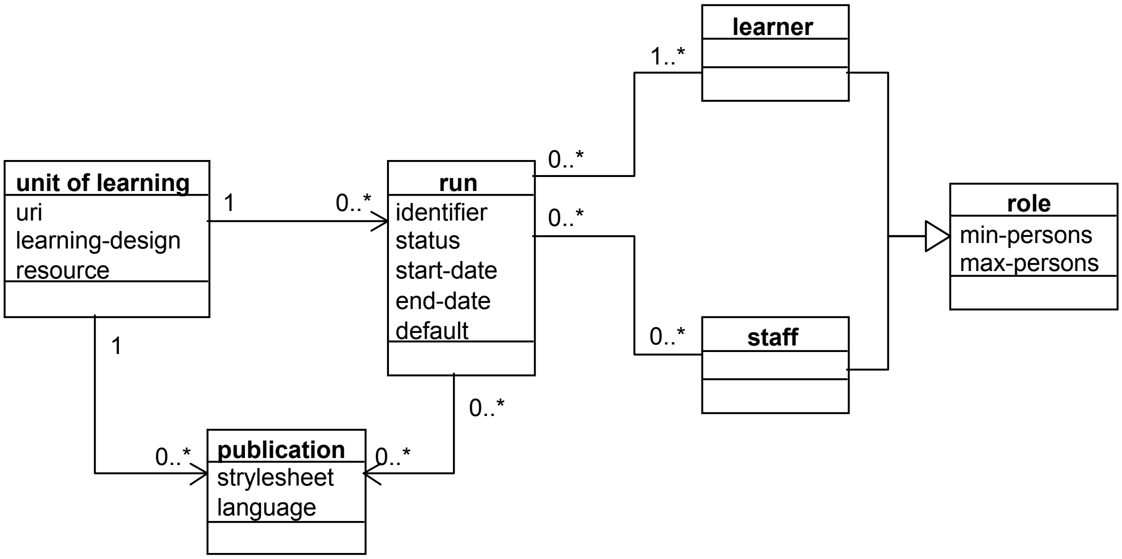

Figure 2. The full UML class diagram for the design.

In summary, the design of an approach to ensuring reproducibility in integrated e-learning systems involves coupling the concept of a unit of learning to that of a run, which links individuals in particular roles to a unit of learning delivered in a given time period.

Implementation

Implementation of the design occurs within the context of the production sub-system of integrated e-learning systems (Koper & Manderveld, 2003). Within this sub-system, a process is introduced, namely, run management.

Our organisation implemented the design in its production processes a number of years ago and is successfully operating with enrolment numbers in excess of 1000 learners per course (i.e. unit of learning), coupled to multiple runs, following different delivery policies, varying from one to ten to hundreds of learners per run.

The run management process is supported by a run tool, the positioning of which is shown in figure 3.

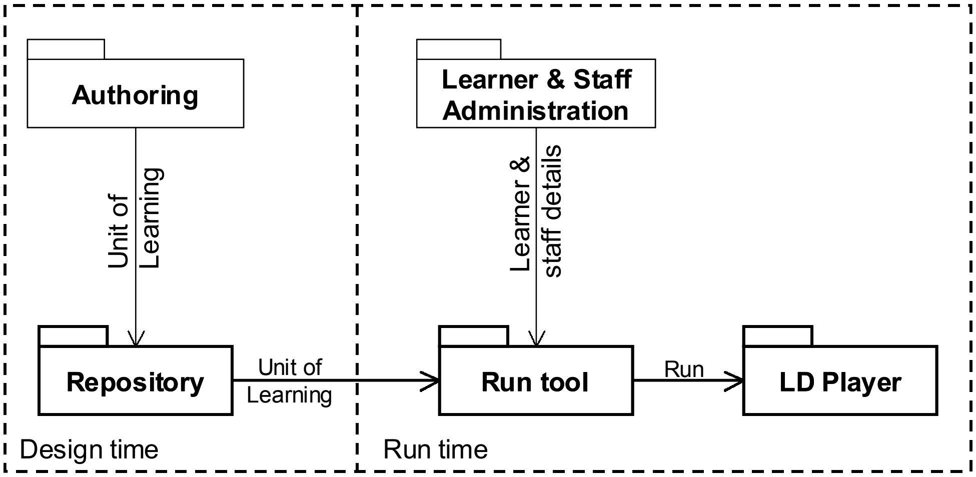

Figure 3: Positioning the run tool in its immediate architectural context

Using the tool, course administrators can access a repository and upload units of learning previously created in an authoring process. The repository is store of the frozen 'parent' units of learning and implements version control mechanisms. Once the abstract course description is available in the run tool, multiple instances can be spawned by linking to learner and staff details pulled in from the appropriate enrolment and administrative systems.

For users (staff or learners) using the player, support is offered in several situations. First, if the user has been assigned to exactly one run for a given unit of learning he or she can be directed to the run (eg through a hyperlink). If the user has been assigned to multiple runs of the same unit of learning, a choice between the available runs is offered. This mechanism can be used, for example, to give learners a choice of starting date, or staff a choice of which run to support when several are running in parallel. If the user has not been assigned to a run but is enrolled for a unit of learning for which a default run is available, the user is assigned to the run automatically. A fourth possibility exists in which the user is not enrolled for a unit of learning (and so is not assigned to a run) but requests access to a run (perhaps through sharing of hyperlinks between learners). In this case enrolment form could be presented to the user, requesting enrolment. This aspect is not implemented in our context, where users are instead denied access and directed to the traditional enrolment process.

The run tool is used as the mechanism by which the status of a run is changed by course administrators. However the opportunities for automation are evident, such as using timed events to move the status from waiting to active at the start of an academic year.

In rounding off this description of the design's implementation, we note that the current implementation in our organisation is based on IMS LD's pre-cursor, EML (Koper et al., 2003). Although EML and IMS LD differ in certain respects, the differences do not alter the requirements or design described in this article, and have only minor repercussions for the implementation.

Discussion

Introducing the unit of learning/run distinction is a simple yet effective way of achieving flexibility of e-learning delivery while preserving efficiency and traceability of administration. The distinction mirrors that used in the book and record publishing industries where similar production and delivery economics apply.

Returning to the example delivery policies outlined in the requirements analysis, we outline how each is addressed by the design:

a course is run once only (then discontinued), with a single set of learners. The unit of learning which represents the course in the abstract is mapped to a single run which is delivered once only.
a course is run for several sets of learners for logistical reasons, such as staff resource limitations, physical room constraints or marketing purposes. *Here, the ESP creates as many runs as needed from the single parent unit of learning given the specific constraints, and at the times the runs are needed.*
a course is run for (possibly several sets of) learners and the learners are divided into groups on pedagogical grounds. This example is addressed in a similar manner to the previous one, with the ESP assigning learners to groups (for example, teams) used in the learning design.
a course is run only when, but as soon as, there are enough enrolled learners. This is a slightly more complex situation but one which underlines the power of the approach described here. Runs can be created by ESPs with appropriate constraints on "min-persons" and given the status of active. This means that although active, the run will not start until the constraints are met. Learners may enrol at any time and are placed into a queue until sufficient numbers are gathered, at which point an alert is issued to learners and staff that learning can begin (the queue can of course be monitored to help with staff planning). Note that runs will continue to be generated from the pool of enrolled learners each time the constraints are met. For example, if "min-persons=50", then as soon as 50 learners enrol, a run will start, and as soon as the 100th learner enrols, a second run commences.
a course is run for each individual learner as soon as the individual's enrolment has been finalised. *Here a single, constraint-free run is created to which enrolled individuals can be directed.*

Archiving and version management run across these examples. The unit of learning which is the basis for the each of the runs remains frozen in the repository, and the link between individuals and a uniquely identified run of a particular unit of learning is logged in learner administration systems.

Turning to the issue of making minor modifications to runs which are in progress, the link between a unit of learning and its runs is maintained, making it is possible to apply minor modifications across all runs in one action.

The approach also opens new avenues when used together with linking technologies. A unit of learning can contain resources, and indeed other units of learning, either directly in the content package or by reference using URIs. The use of referencing rather than direct inclusion in a content package makes it possible to deliver units of learning while referenced subcomponents are still under development—a link is created in a unit of learning to a location in which another resource will be placed when completed. The unit of learning can be frozen, since it will not be modified, and runs can be spawned to reach the market before production has completely finished.

The design meets the four requirements outlined in the Requirements Analysis and has been implemented in a production level environment. With respect to its wider applicability, we note two current developments in the e-learning specification and standardisation world

First, the Internet2 Middleware Architecture Committee for Education (2003c) 'course data elements project' is creating a standard data element syntax to describe courses and is exploring the relationships between courses and classes (using the word 'class' in its educational rather than computational science sense) having identified the importance of the distinction between the generic and the specific. In contrast, the Project Editor's Draft of the ISO/IEC 19780, Collaborative Technology — Collaborative Workplace — Learner to Learner Interaction Scheme standard (ISO/IEC JTC1 SC36, 2003), does not appear draw the distinction between the two levels, even though its goal is reproducibility—"The purpose of this standard is to prepare a system independent description method of collaborative learning entity so that such entity can be used repeatedly to realize the same learning environment for multiple collaborative learning systems".

These initiatives illustrate the relevance and timeliness of our contribution, which we believe to be fundamental to realising e-learning's economic promise, yet straightforward to implement.

Acknowledgements

The authors wish to thank the management and staff of the Schloss Dagstuhl International Conference and Research Center for Computer Science for providing a pleasant, stimulating and well organised environment for the writing of this article.

References

IMS Content Packaging Specification (2003a). [On-line]. Available: http://www.imsglobal.org/content/packaging/index.cfm

IMS Learning Design Specification (2003b). [On-line]. Available: http://www.imsglobal.org/learningdesign/index.cfm

Middleware Architecture Committee for Education (MACE) (2003c). [On-line]. Available: http://middleware.internet2.edu/MACE/

Abdallah, R., El Hajj, A., Benzekri, A., & Moukarzel, I. (2002). On the Improvement of course interoperability in E-Learning Models. In *International Conference on Engineering Education 2002*.

Brusilovsky, P. & Miller, P. (2001). Course Delivery Systems for the Virtual University. In F.T.Tschang & T. Della Senta (Eds.), *Access to Knowledge: New Information Technologies and the Emergence of the Virtual University* (pp. 167-206). Amsterdam: Elsevier Science and International Association of Universities.

Canadian Department of National Defence (2003). White paper: SCORM Dynamic Appearance Model. [On-line]. Available: http://www.online-learning.com/papers/SCORMModel.pdf

Daniel, J. (1998). *Mega-universities and Knowledge Media.* RoutledgeFalmer.

Douglas, I. (2003). Instructional Design Based On Reusable Learning Objects: Applying Lessons Of Object-Oriented Software Engineering To Learning Systems Design. In *Proceedings of the 31st ASEE/IEEE Frontiers in Education Conference* IEEE.

Downes, S. (2001). Learning Objects: Resources For Distance Education Worldwide. *International Review of Research in Open and Distance Learning, 2.*

Friesen, N. (2001). What are Educational Objects? *Interactive Learning Environments, 9,* 219-230.

ISO/IEC JTC1 SC36 (2003). Project Editor's Draft: ISO/IEC 19780, Collaborative Technology — Collaborative Workplace — Learner to Learner Interaction Scheme. [On-line]. Available: http://collab-tech.jtc1sc36.org/doc/SC36_WG2_N0051.pdf

Koper, R. & Manderveld, J. (2003). Educational Modelling Language: Modelling reusable, interoperable, rich and personalised units of learning. *The British Journal of Educational Technology.*

Koper, R. (2003). Learning Technologies in eLearning: An Integrated Domain Model. In W.Jochems, J. van Merrienboer, & R. Koper (Eds.), *Integrated E-Learning: Implications for Pedagogy, Technology and Organization,* Kogan Page.

Molyneux, S. (2000, July). Where technology enables. *Open Learning Today - The Journal of the British Association for Open Learning*.

Morgan, G. (2003). *Faculty Use of Course Management Systems: Key Findings* EDUCAUSE Center for Applied Research.

Nulden, U. (2001). e-ducation: research and practice. *Journal of Computer Assisted Learning, 17,* 363-375.

Permanand, M. & Brooks C. (2003). Engineering a Future for Web-Based Learning Objects. In *Lecture Notes in Computer Science Volume 2722 / 2003* (pp. 120-123). Heidelberg: Springer-Verlag.

Porter, D. (2001). Object Lessons From the Web. In G.Farrell (Ed.), *The Changing Faces of Virtual Education* (pp. 47-69). Vancouver: The Commonwealth of Learning.

Sloep, P. (2003). The Language of Flexible Reuse: Reuse, Portability and Interoperability of Learning Content or Why an Educational Modelling Language. In R.McGreal (Ed.), *Online Education Using learning Objects* London: Kogan Page.

Virvou, M. & Tsiriga, V. (2001). An object-oriented software life-cycle of an intelligent tutoring system. *Journal of Computer Assisted Learning, 17,* 200-205.

Widya, I., Volman, C., Pokraev, S., De Diana, I., & Michiels, E. (2002). Enterprise Modelling for an Educational Information Infrastructure. In J.Filipe, B.Sharp, & P.Miranda (Eds.), *Enterprise Information Systems III* (pp. 231-239). Kluwer Academic Publishers.