

# DEVELOPING ADVANCED UNITS OF LEARNING USING IMS LEARNING DESIGN LEVEL B

Rob Koper, Daniel Burgos  
Educational Technology Expertise Center (OTEC)  
Open University of the Netherlands  
Valkenburgerweg 177; PO Box 2960  
6401 DL Heerlen; The Netherlands  
email: [rob.koper@ou.nl](mailto:rob.koper@ou.nl)

## Abstract

IMS Learning Design (LD) is an open specification, released in 2003, to support the interoperability of advanced pedagogical designs in e-learning courses and other 'units of learning'. The specification supports three levels: level A is the basic level, level B adds 'properties, global elements, monitor services and conditions' and level C adds 'notifications'. Current tools for LD support level A of the specification. Some new tools are exploring the incorporation of level B, but there is still a lot of unexplored territory in this area. Also the documentation that is available today explains in detail the technical differences between level A and B, but not functionally. This *tutorial paper* introduces the possibilities of using IMS Learning Design level B to develop digital courses that support: collaborative learning, adaptive learning and personalisation, conditional text, runtime tracking, new forms of assessment and the modelling of ePortfolio's.

## Key Words

IMS Learning Design; Adaptive Learning;  
Collaborative Learning; Assessment;  
Instructional Design

## 1. Introduction

IMS Learning Design (LD) is an open specification that is used to code a wide variety of digital courses and other units of learning in a formal, semantic, interoperable and machine readable way [1]. It is especially strong in the support for the wide range of modern pedagogical approaches that are used today, e.g. active learning, collaborative learning, adaptive learning, competency based learning [2]. The LD specification became available in 2003 and currently a lot of projects are building tools for it or are exploring its use by developing concrete

examples. The current state-of-the-art is described in a recent book about LD [3] and the EU UNFOLD project is co-ordinating the activities in the field and stimulates the wider adoption of LD [4]. Part of the UNFOLD project are the Communities of Practice and digital courses to learn about LD [5]. Most of the current tools and example courses are supporting level A of the specification. This level provides all of the basic structures to model courses, but not the more advanced mechanisms, like properties and conditions (level B) and notifications (level C). The use of LD level B provides complete new possibilities for the design of digital courses, but it is quite hard to understand from the existing specifications and documents what these new possibilities are and how these can be used in practice. This is the reason why this tutorial paper explains some of the basic possibilities of level B and shows how this can be modelled with properties and conditions.

The paper starts with introducing the core concepts of LD level B: what are properties, global elements, monitor services and conditions. This will be followed by a section that explains four different core examples of the use of LD level B: collaborative learning, adaptive learning and personalisation, conditional text & runtime tracking, and ePortfolio's & new forms of assessment. The paper ends with some conclusions about the use of LD level B.

## 2. IMS Learning Design Level B

This section provides a short introduction to the core elements of IMS LD level B. It supposes that you have direct access to the IMS LD Information Model [1] as a reference guide.

Learning Design Level B adds several elements

to the level A model of the specification. It enables personalization, more elaborate sequencing and interactions based on learner portfolios. It can be used to direct the learning activities as well as to record learning outcomes.

LD level B is related to two of the eight design requirements of LD (see the information model [1]):

### *R2. Pedagogical Flexibility:*

The specification must be able to express the pedagogical meaning and functionality of the different data elements within the context of a unit of learning. It must be flexible in the description of all different kinds of pedagogies and not prescribe any specific pedagogical approach.

Level A only supports a restricted set of pedagogical approaches. Level B is needed to support a wide variety of more complex, adaptive personalized or assessment based approaches. Some of these will be addressed in this paper.

### *R3. Personalization:*

The specification must be able to describe personalization aspects within a learning design, so that the content and activities within a unit of learning can be adapted based on the preferences, portfolio, pre-knowledge, educational needs, and situational circumstances of users. In addition, the control over the adaptation process must be given, as desired, to the student, a staff member, the computer, and/or the designer.

Level A has only very limited support for personalization and adaptation. For a real adaptation, at least level B is needed.

The specific elements that level B adds to level A are:

1. *Properties* to store information from users and groups of users.
2. *Global elements* to set and view the information stored in properties. Properties can be read by the user himself or by others (fellow learners or tutors).
3. *Monitor service* to read the properties of other persons or yourself.
4. *Conditions* that work on property values to adapt or personalize a variety of elements

within or outside the learning design.

These additional elements will now be explained one by one, without going in too much detail.

## **2.1 Properties**

Properties are variables that can hold a variety of information types, e.g. texts, booleans, numeric values, files. Properties are *declared* in the LD manifest file and the property values are *set* and *viewed* using XHTML resources. These resources are of type 'imsldcontent' and are included in the content package (the 'unit of learning'). An XHTML resource includes so-called 'global elements' that set and view property values. Examples will be provided in section 3.

LD defines five types of properties:

1. *Local properties*. This property is only accessible within the context of a unit of learning. It has exactly *one* value that is the same for all users. An example is: teacher provides a common introduction to the course (or himself) by typing some text into a property.
2. *Local personal properties*. These are also only accessible within a unit of learning, but every person has a different value for the property. An example is: students each individually have to upload a report for review.
3. *Local role property*. Only accessible within a unit of learning. Contains a different value for every role in the unit of learning. For example: two groups of students (= 2 roles) each have to create a proposal for the solution of a problem.
4. *Global personal properties*. These are declared in *one* separate unit of learning and the value can be set and viewed in *every* unit of learning by the person who owns the data. Each person has separate values for the properties. For example portfolio data like: name, address, email, scores, competence levels, products, etc. of a student.
5. *Global properties*. These are declared in *one* separate unit of learning and the (single) value can be set and viewed in *every* unit of learning by all persons who have access. An example is: container for regulations and

policies of an institution.

Properties can be grouped to a record using the *property-group* element. Property-groups can also be set and viewed using global elements.

## 2.2 Global Elements

Global elements are LD elements that can be embedded in external XML or XHTML resources by 'namespacing' them in (see examples). These resources must have the resource type 'imsldcontent' to inform the runtime engine that it has to look for global elements in its content. Global elements are used to set and view property values or the values of the properties that are sequenced in property groups.

Viewing and setting property values can be done in the context of a text, a form, a graphic, or any other XML based resource. This dynamically integrates the learning design with the learning resources. One of the results is that XML resources can be adapted during runtime, e.g. can include texts that are the values of properties.

The global elements are:

1. *view-property* and *view-property-group*. To show the value of a property(-group) at the position where the element is encountered in the XML file.
2. *set-property* and *set-property-group*. When a runtime engine encounters these elements it will generate an input control at its position. The type of input control depends on the data type. For example, a text box is presented when the property is of type 'text', a combo box is presented when the user has to select from a list of choices, a file upload control is presented when a property contains a file.

The 'set-property' command has an optional attribute 'max-transactions' that can be used to control whether a student may upload a property value more than once or not.

Because properties are used and set within the external XML resources, level B authoring environments need to include an X(HT)ML resource editor that allows the easy inclusion of global elements in the context of X(HT)ML elements.

## 2.3 Monitoring

As has been discussed, the local and global personal properties have a different value for each individual person. When viewing or setting these properties it must be specified which property values have to be viewed or set: the property of the person himself or the properties of other persons within the same role.

LD has two mechanisms to specify this:

1. Each global element has the attribute 'property-of' to specify whether it is the value of the property of the user himself ("self"), or that it is the value of other persons in a role ("supported persons").
2. When using this last option, also the *role* must be specified using the element 'monitor' within the environment of an activity.

Using these two constructs it is possible to view and set the properties of your self, and to view and set the properties of others (given adequate access rights that are handled in the runtime system). This will be further illustrated in the examples.

## 2.4 Conditions

Conditions are the basic mechanism to specify the dynamic behaviours in the unit of learning. Conditions are 'if – then – else rules' within the IMS manifest file to adapt or personalize the activities or resources or to calculate property values. It includes an extensive and expandable expression language to specify the 'if' statement. When a person performs a request to the runtime engine, the relevant conditions are evaluated. When the expression of a condition is true, the rule fires the 'then' part, otherwise it will fire the 'else' part (when present, when the else part is not present, it does nothing).

Basically, all the 'if' expressions read the property values that are defined in the learning design, or are added by the runtime engine (e.g. date/time that the run of a unit of learning has started).

Using this mechanism, the personalisation or adaptation results can be completely different for different persons using the same unit of learning at the same time.

### 3. Examples

In this section we will provide examples of the use of LD at level B. We will specifically concentrate on some modern pedagogical approaches: collaborative learning, adaptive learning, personalization, conditional text, runtime tracking, ePortfolio's and new forms of assessment. The examples include some pseudo XML LD code that is abstracted in such a way that it concentrates on the main elements and attributes. Superfluous details and additional tags have been ruled out in order to get a better presentation. The examples that are based on real courses are all available at [5] and run in the CopperCore Learning Design engine [6].

#### 3.1 Collaborative learning

Collaborative learning [7] is a pedagogical approach in which students at various competency levels work together in groups toward a common learning objective. The group size, the type of interaction between the participants and the type of learning objective may vary from situation to situation [8]. Collaborative learning increases interest among participants, facilitates the cognitive representation and far transfer of knowledge and stimulates critical thinking. Learning environments that support collaborative learning should at least facilitate: the grouping of students, the support for discussions within the groups and the monitoring of your own progress and that of the group. Additionally a learning environment can include specific collaborative tools (e.g. wiki's for collaborative writing or problem solving) and facilities for teachers to monitor and support the group in an efficient way. For example, a learning environment may offer facilities for teachers to monitor the progress of the students, analyse the students' contributions and provide proper feedback to the students individually or to the group. LD level A provides the basic mechanisms to setup and rearrange learner groups in runtime (through the 'role' attribute 'create-new'), to define communication services (with the 'forums' services) or to connect to collaborative tools (with 'learning objects/tools' ). In addition, it is possible with LD level B to include information at runtime (e.g. a teachers introduction to the course) or to

'monitor' the performance of individual learners using the 'monitor' service. With the monitor service, all types of properties can be viewed: properties of the learner himself, properties of others in the same group, or properties of other individuals or groups. To establish this facility one has to define a set of properties in the LD manifest file and create a separate XHTML file with global (view-) properties to read and present the property values with the monitor service.

We will now present an example. First we define a local personal property in the LD manifest file.

```
<locpers-property identifier="Personal-learning-objective">
  <title>Which learning objective do you want to
  achieve with this course?</title>
  <datatype="text"/>
  <initial-value>write your objective here</initial-
  value>
</locpers-property>
```

The property 'Personal-learning-objective' has a title ('Which learning objective...') and is of type 'text'. The (optional) initial value is set to 'write your ..'. The property will store the learning objective of each individual student.

The next step is that we define an XHTML document where the user can input the information, using the global element 'set-property'. A fragment of such a file is the following:

```
<p>Specify your personal goals for this course
below:</p>
<ld:set-property ref="Personal-learning-objective"/>
```

When a user has access to this page, the runtime engine will present the user with the question ('Specify your ..') and the set-property will be translated to an input form where the user can enter the textual information. The runtime engine will store this information in the property of each individual student.

To view the value of the property, another XHTML file can be created that includes the global element 'view-property':

```
<p>Your learning goals are:</p>
<ld:view-property property-of="self" ref="Personal-
learning-objective"/>
```

The attribute 'property-of' is set to 'self' to specify that the value of your own property must be viewed (instead of the values of all the participants). This XHTML file can be connected to any item element in the environment (most likely a learning object or the monitoring service). However, when you want to view the properties of other persons in a group (role), then you need the monitor service to provide the context for the property, i.e. the properties of persons in which role do you want to retrieve?

For example, to view the learning objectives of fellow students in the group/role 'Participant' the value of the attribute 'property-of' in the XHTML file should be set to "supported-persons" instead of "self".

```
<ld:view-property property-of="supported-person"
ref="Personal-learning-objective"/>
```

To identify of which supported persons the property values should be viewed, a monitor service is specified in the environment:

```
<environment>
<service>
<monitor>
<role-ref ref="Participant"/>
<title>View the learning objectives of the other
participants: </title>
<item identifierref="Personal-learning-objective"/>
</monitor>
</service>
</environment>
```

Every person who has access to this monitor object (e.g. tutor or participants) can now view the learning objectives as they are specified by all the participants. This mechanism can be used in a variety of ways in the learning design. One example is that the participants discuss the variety of learning objectives and agree upon a common learning objective. Such an objective can be stored in a separate local role property with *one* value per role/group. The value can be retrieved in any other XHTML resource by using the 'view-property' element.

### 3.2 Adaptive learning and personalization

Adaptive learning is a pedagogical approach that aims to personalize the instruction by providing each individual learner a set of learning

activities and resources that fits the individuals' learner properties such as the personal learning objectives, prior knowledge or situational circumstances [7]. This is opposed to static learning systems that present the same material to every user in the same way and order. Personalization is the possibility that a learning environment provides to customize certain aspects according to the users' preferences (e.g. text size, language, etc.).

To illustrate these mechanisms of adaptation and personalisation we use the LD course 'Learning to listen to Jazz' (available at [5]) initially developed in EML and later adapted to IMS LD. In this unit of learning a student can follow a course about Jazz and can choose two different learning routes, a thematic or a historical route. The programme presents an advice for the best route. This advice is based on the results of an assessment of the prior knowledge of the student and on an assessment of the learning style of the student. However, the student is free to follow or not to follow the advice.

In the Jazz course, a property called 'choose-route' is defined to store whether the user has chosen one of the two routes. Each route is described in a separate 'activity-structure', 'AS-historic' and 'AS-thematic'. A set of conditions are defined to specify which route should be shown or hidden depending on the value of this property. First of all, when the property has 'no-value' than both routes should be hidden:

```
<conditions>
<if>
<no-value>
<property-ref ref="choose-route"/>
</no-value>
</if>
<then>
<hide>
<activity-structure-ref ref="thematic"/>
<activity-structure-ref ref="historical"/>
</hide>
</then>
```

When the student has selected one of the two routes, the property is set to historical or thematic (through a combo box in the user interface). If the user selects the option 'thematic', the related structure 'thematic' is shown and the non-related structure 'historical' is hidden. The same things happens the other

way around:

```

<if>
  <is>
    <property-ref ref="choose-route"/>
    <property-value>thematic</property-value>
  </is>
</if>
<then>
  <show>
    <activity-structure-ref ref="thematic"/>
  </show>
  <hide>
    <activity-structure-ref ref="historical"/>
  </hide>
</then>

```

The two different activity-structures can contain a different ordering of the same learning activities, but also a complete different set of learning activities. It should be noted that also complete 'plays' instead of activity structures can be selected according to a specific user profile. This can be used for instance to provide to forms of education based on the same resources, but for different settings (e.g. for classroom teaching and for distance learning).

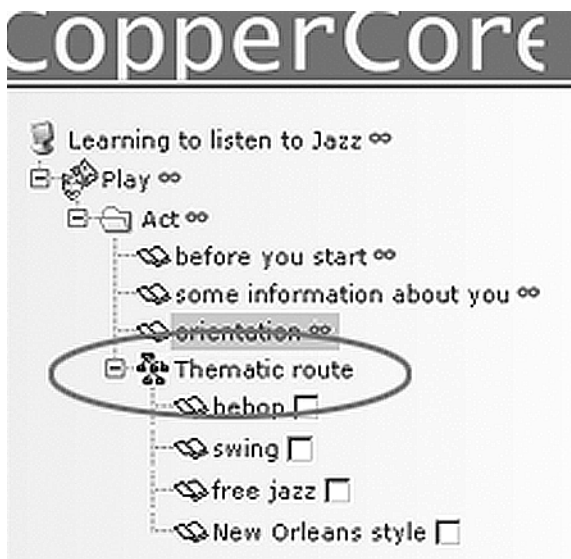
To illustrate personalization we will now look at the question how we can access the personal preferences of a student. In the previous section we showed how to define and fill a single property. Another possibility is to group several properties under a single name to allow for easy operation. For instance

Take for instance the following two global personal properties:

```

<globpers-property identifier="name">
  <title>your name</title>
  <datatype datatype="string"/>
</globpers-property>
<globpers-property identifier="age">
  <title>your age</title>
  <datatype datatype="integer"/>
  <initial-value>0</initial-value>
</globpers-property>

```



We can group both properties with a 'property-group':

```

<property-group identifier="profile">
  <title>personal data</title>
  <property-ref ref="name"/>
  <property-ref ref="age"/>
</property-group>

```

The information can now be requested ('set-property-group') and shown ('view-property-group') with a single reference to the group identifier:

```

<Id:set-property-group ref="profile" property-of="self"/>
<Id:view-property-group property-of="self" ref="profile" />

```



Figure 1. The Thematic and Historical routes

These data can be used in conditions to do a variety of things, like: showing or hiding specific texts (using 'class' attributes in XHTML texts), showing or hiding specific activities,

showing or hiding specific resources, showing or hiding specific activity structures or plays.

It should be noted that the adaptation and personalization examples are very simple. With LD level B an almost endless set of adaptation and personalization principles can be designed, depending on the designers needs. For instance, some approaches to adaptive learning use learner models. It is possible to use properties to represent a user model (e.g. a learning style property, learning objectives, competency levels on different competencies, language preferences, etc.). Conditions are specified in such a way that they only show an activity when the expression in the 'if' part of the condition matches the user model. (eg. if [user speaks English] and [user needs an quick introduction into psychology] then show [activity 'introduction psychology']).

These examples of adaptation are very 'design driven': the designer foresees all the possibilities and defines adaptation rules based on user models. Other approaches to adaptive learning are less design driven. In these approaches a user can select from all the activities himself (an activity-structure of type 'selection'). This can be modelled with LD level A. With level B it is possible to extend this to let the user select how much support he or she wants.

### 3.3 Conditional Text and runtime tracking

With LD level B elements it is possible to model surveys, questionnaires, etc. when needed (however for interoperable tests, the IMS QTI specification [10] should be used preferably). In this section we illustrate how we model some questions and provide dynamic feedback. Furthermore we will explain how to track the results at runtime.

In order to illustrate this section we take the example 'GeoQuiz' (available at [5]). In this unit of learning a student answers a set of five questions. Feedback is given to the user, depending on the selected responses. When all the questions are answered a numeric average grade is provided and a final remark based on it. Based on the value of a property it is possible to specify that certain parts within an XHTML resource can be shown or hidden (e.g. additional explanations, context depended examples, feedback, etc.).

The mechanism is basically the same as the showing and hiding of structures within the learning design. However, the major difference is that the runtime engine now influences external resources instead of structures within the IMS manifest file. For instance, a user answers a question. When the answer (Answer1) is response "c" than the answer is right and some feedback should be given:

```
<if>
  <is>
    <property-ref ref="Answer1"/>
    <property-value>c</property-value>
  </is>
</if>
<then>
  <hide>
    < class="Feedback_Wrong"/>
  </hide>
  <show>
    <class="Feedback _Right"/>
  </show>
</then>
```

Structures in the XHTML file (e.g. using the <div class=""/> element) with the class attribute 'Feedback\_Right' are shown and structures with 'Feedback\_Wrong' are hidden. In the external XHTML file, the value of the property 'Answer1' is set and the class elements are defined as follows:

a) The question is modelled as follows:

```
<p>Blended Learning is:</p>
<p>a. a kind of cocktail mixed with a blender</p>
<p>b. a mixture of different pedagogical
approaches</p>
<p>c. a mix between f2f and online learning</p>
<p>Answer:
<set-property ref="Answer1" of="self"/></p>
```

b) the classes are defined as follows:

```
<div class="Answer1_Wrong">
  <p><view-property ref="Answer1"/> is not
right</p>
</div>
<div class="Answer1_Right">
  <p>Congratulations!</p>
  
</div>
```

In addition we can make arithmetic calculations with the stored property values and provide contextual feedback based on these calculations.

For instance, when we have two questions with two answers and two possible values for each answer, 0 and 100, we could:

- a) define every property in the following way:

```
<locpers-property identifier="QuestionTrue1">
  <datatype datatype="integer"/>
  <initial-value>0</initial-value>
</locpers-property>
```

- b) assign a value based on a specific answer

```
<if>
  <is>
    <property-ref ref="Answer1"/>
    <property-value>c</property-value>
  </is>
</if>
<then>
  <change-property-value>
    <property-ref ref="QuestionTrue1"/>
    <property-value>100</property-value>
  </change-property-value>
</then>
<else>
  <change-property-value>
    <property-ref ref="QuestionTrue1"/>
    <property-value>0</property-value>
  </change-property-value>
</else>
```

- c) calculate a simple average with the two answers as arguments:

```
<change-property-value>
  <property-ref ref="sum"/>
  <property-value>
    <calculate>
      <divide>
        <sum>
          <property-ref ref="QuestionTrue1"/>
          <property-ref ref="QuestionTrue2"/>
        </sum>
        <property-value>2</property-value>
      </divide>
    </calculate>
  </property-value>
</change-property-value>
```

### 3.4 ePortfolios and new forms of assessment

Besides the well known tests (multiple choice tests, essays, etc.) new forms of assessment have been developed in the last decade. In the new learning approach assessment is more integrated

in learning and instruction and measures the complex competencies of students more than local simple skills or knowledge. New types of assessment are developed, like peer assessment or competence assessment [11]. LD is designed in such a way that it can include the classical forms of testing (by using the IMS QTI within the LD context), but also the newer forms of assessment.

Most of these new assessment forms can be modelled with LD level B properties, conditions, monitor services and global elements. The basic examples have already been provided in the previous sections. However, we will now provide a short example to illustrate how to include IMS QTI tests within a learning design

```
<environment identifier="Env-1">
  <title>A test linking QTI and IMS LD</title>
  <title>First question</title>
  <item identifier="item1" identifierref="QTI-item1"/>
  </learning-object>
</environment>
```

```
<resource identifier="QTI-item1" type="imsqti">
  <file href="question1.xml"/>
</resource>
```

The QTI test item can set a property value that can be used within the LD context. This is described in detail at [10]. The principle is that when property identifiers and variable names are declared to be lexically identical at design time (i.e. in IMS LD-based and IMS QTI-based XML), they are considered to be a shared variable in run-time software environments which involve IMS LD and IMS QTI-based processes.

Besides assessment, ePortfolio's are used in education to store the results of assessments along with products that are produced during learning [12]. A portfolio saves and shares standardized sets of data externally to applications and units of learning, allowing its use at different places, with different systems and within different units of learning. The global personal properties in IMS LD are used to define portfolio data. To setup portfolio properties (as any global property) we need to create a special unit of learning that is only used to declare the global portfolio properties. It contains expressions like the following:

```

<globbers-property identifier="competency-level-
language">
  <global-definition uri="competency-level-language-
english">
    <title>The competency level of the user in English
classified according to the European Framework for
Language Competencies</title>
    <datatype datatype="integer"/>
  </global-definition>
</globbers-property>
<globbers-property identifier="example-report-
language-english">
  <global-definition uri="example-report-language">
    <title>The most recent written English text"</title>
    <datatype datatype="file"/>
  </global-definition>
</globbers-property>

```

The second property is a file that can be uploaded to the portfolio in any XHTML resource assessable to the user that contains the following fragment:

```

<div class="upload-file">
  <p>Upload the most current text you have
written in English to provide an example of your
English writing skills:</p>
  <Id:set-property ref="example-report-language-
english"/>
</div>

```

#### 4. Conclusion

In this tutorial paper we presented the basic constructs of IMS Learning Design level B. We discussed the elements that are added to the level A model: properties, global elements, monitor service and conditions. These elements together are responsible for the modelling of dynamic units of learning. With level A, there modelling power is restricted to standard sequences or selections of learning activities for multiple users. With level B it is – among other things - possible:

- to adapt learning activities or external resources to a users needs, prior knowledge and situational circumstances,
- to create digital portfolio's and to use the information in the portfolio to provide a more personalised learning environment,
- to include advanced sequencing of learning activities. Sequencing in level A is restricted to fixed sequencing and the inclusion of IMS Simple Sequencing. Level B can extend this by using the property and conditions mechanisms, e.g. to sequence activities based

on advanced user models.

- to model new and classical forms of assessment that are integrated into the learning process.
- to change the content of a learning activities or resources at runtime (through properties),
- to ask input from users and to show the results to selected users to support all kinds of collaborative processes in learning.
- to calculate, aggregate and present numeric data (e.g. test data).

LD Level C will even extend this mechanism by allowing even more advanced designs that are based on dynamic task selection, and by providing workflow mechanisms that can make the work of teachers and students more efficient.

To conclude this paper a general *comment* must be made. The examples provided above (and IMS LD in general) deal with concepts that are modelled during *design time*. This is only a part of the story when facilitating learning. Learning takes place at runtime, the design mechanisms will set-up the environment and will automate parts of it, depending on the design. So, the design provides the facilities and constraints for the runtime learning activities. In runtime the users communicate, change property values, regroup each other, use collaborative tools, have ad hoc questions, etc. This means that a unit of learning coded with LD should run in a learning management environment that offers specific applications and services that are not part of the design itself but generic to the learning management system. Designers should be aware of this principle by not wanting to include every possible detail in the design, but by designing a learning environment that facilitates a variety of choices at runtime. Depending on its use, a unit of learning can be very generic, i.e. similar to the design of one of the current learning management systems (e.g. the topic view and the social view of Moodle can be seen as two very open learning designs that are customized and used by teachers and learners in runtime). A unit of learning can also be modelled very specific, e.g. a distance teaching course to introduce statistical methods that are used by large numbers of students with a limited amount of tutoring.

Besides the question whether the initial design

is open or more restricted, users must be facilitated to change the design of a unit of learning when it seems necessary during runtime. This mechanism is almost never provided in the current learning management systems (most of the changes involves additional programming), but is provided by LD runtime engines like CopperCore [6]: existing designs can be changed, and runtime environments can be updated according to the new design.

## References

- [1] IMS Learning Design. Information Model, Best Practice and Implementation Guide, XML Binding, Schemas. Version 1.0 Final Specification. (Boston: IMS Global Learning Consortium, 2003). Retrieved April 14th, 2005 from <http://www.imsglobal.org/content/learningdesign/>
- [2] Van Es, R., & Koper, R. Testing the pedagogical expressiveness of LD. (submitted paper). Retrieved April 14th, 2005 from <http://hdl.handle.net/1820/305>.
- [3] Koper, R., Tattersall, C. Learning Design: a handbook on modelling and delivering networked education and training. (Heidelberg: Springer, 2005).
- [4] <http://www.unfold-project.net>
- [5] <http://moodle.learningnetworks.org>
- [6] Vogten, H., Martens, H., CopperCore 2.2.2 (Heerlen: OUNL, 2005). Retrieved April 14th, 2005 from [www.coppercore.org](http://www.coppercore.org).
- [7] Boticario, J. Santos, O., Barrera, C. Gaudio, E., Hernandez, F. Rodriguez, A., Van Rosmalen, P. Koper, R. Defining adaptive learning design templates for combining design and runtime adaptation in aLFanet. Retrieved April 14th, 2005 from <http://hdl.handle.net/1820/210>
- [8] Dillenbourg P. What do you mean by collaborative learning?. In P. Dillenbourg (Ed) *Collaborative-learning: Cognitive and Computational Approaches*. (Oxford: Elsevier, 1999) 1-19.
- [9] available at <http://moodle.learningnetworks.org>
- [10] IMS Question and Test Interoperability Specification. (Boston: IMS Global Learning Consortium, 2004). Retrieved April 14th, 2005 from <http://www.imsglobal.org/question/index.cfm>
- [11] Joosten – ten Brinke, D., Van Bruggen, J., Hermans, H., Burgers, J., Giesbers, B. & Koper, R. A Conceptual Model for Assessment: Re-use of Traditional and New Types of Assessment. (submitted paper).
- [12] IMS ePortfolio (Boston: IMS Global Learning Consortium, 2004). Retrieved April 14th, 2005 from <http://www.imsglobal.org/ep/index.html>