Building Adaptive Game-Based Learning Resources: The Marriage of IMS Learning Design and <e-Adventure>

Daniel Burgos[1], Pablo Moreno[2], José Luis Sierra[2], Baltasar Fernández-Manjón[2], Marcus Specht[1], Rob Koper[1]

[1]Open University of The Netherlands
Educational Technology Expertise Centre (OTEC)
Valkenburgerweg, 177
6419AT Heerlen The Netherlands
{daniel.burgos;marcus.specht;rob.koper}@ou.nl

[2]Complutense University of Madrid
Department of Software Engineering and Artificial Intelligence
Profesor José Garcia Santesmases s/n
28040 Madrid, Spain
{pablom;jlsierra;balta}@fdi.ucm.es

Abstract

IMS Learning Design (IMS-LD) is a specification to create Units of Learning (UoLs) which express a certain pedagogical model or strategy (e.g., adaptive learning with games). However, the authoring process of a UoL remains difficult due to the lack of high-level authoring tools for IMS-LD, even more so when we focus on specific topics like educational eGames. However, we can use external tools which are not specifically IMS-LD oriented. In this case, the main challenge is the integration between these external resources developed with other technologies and the personalized learning experience of an IMS-LD UoL. In this paper, we use the <e-Adventure> Project to develop conversational games that are engaged with IMS-LD UoLs to improve personalized learning. The main contribution of this setting is the integration of both parts, the IMS-LD specification and <e-Adventure>, and the communication that enables a mutual influence of the adaptive learning experience.

Keywords

IMS Learning Design; <e-Adventure>; communication layer; eGames; integration; adaptive learning; Game-based learning

1. Introduction

1.1 . Adaptive learning and educational electronic games

Educational electronic games (or simply eGames) can provide adaptive learning and fully personalized itineraries, in addition to the motivational enhancements of the learning process [1-3]. EGames clearly improve motivation and involvement, which results in a deeper learning experience [4]. When thinking in terms of adaptive learning, it is remarkable how adaptation in games is a very common feature (typically based on a number of inputs, i.e., the user's behavior, performance, previous knowledge or personal decisions).

Within eGames, we distinguish different subgroups or genres, one of them being conversational games. These specific games have several properties that are worth examining: they are easy to understand and to model, they are well established and have a long tradition in the game industry, and they provide an engaging and challenging experience where the content and the player's performance have a key role in the flow. Additionally they provide a long list of interesting pedagogical elements: fun, interactivity, problem solving, user involvement, or motivation and creativity, just to mention a few [5, 6]. Finally, they also awake personal feelings and emotions in the players, and provide support for the development of personal skills like focused goals, rules, tasks, affiliation, choice, or lack of adverse consequences in the case of a wrong choice [7].

EGames also provide input, output and feedback in real-time [8], which are used in adaptive learning [9], e.g., choosing the next action to take or the contextualized help provided. In order to achieve the educational objectives, we can use various interactive learning techniques (i.e., learning from mistakes, goal-oriented learning, role playing and constructivist learning [3]) within and/or around the game itself. The main goal is to turn the game into a fully integrated activity within the whole learning process, instead of remaining as an isolated stand-alone resource. In doing so, generic games, as well as specific educational games, can be interwoven with the rest of the learning experience, increasing the educational value [1]. However, we still need an effective integration model to improve the use of games in this way. But achieving an effective integration poses several authoring challenges and we also need development processes to make the authoring process faster and easier for teachers and learning designers when describing personalized learning processes [10]. The joint use of IMS Learning Design (IMS-LD) and <e-Adventure> addresses these issues.

1.2 . IMS Learning Design and <e-Adventure>

IMS-LD [11] provides a flexible specification for describing pedagogical models. One of the main objectives of this specification is to provide a personalized learning experience. It also enables sophisticated adaptive behaviours, such as adaptation of the learning flow, the content or the interface [12]. As in most standards, along with this main educational goal, interoperability and integration of information packages are technical goals.
In order to improve the richness of the learning experience, current research and extensions of IMS-LD are increasingly discussing the integration of external modules. In this sense IMS-LD can be complemented with off-the-shelf components and resources integrated in Units of Learning (UoLs). When this integration is possible, adaptation and learning with IMS-LD can be improved and the authoring process on these topics could become richer and more specific.
In this line, the <e-Adventure> Project has developed a language, an engine and an editor to author conversational eGames that can be integrated with IMS-LD-based learning experiences. The main goal of the <e-Adventure> Project is to apply a documental approach [13] to the development of educational adventure videogames (often also referred as *point and click* adventure games or conversational games). The idea is to allow an author without a strong technical background to produce and maintain an entire game as a document by using an easy-to-understand markup language. This document is then fed to a compiler/interpreter (the <e-Adventure> engine) which produces a fully functional game.
In the forthcoming sections we show how an adaptive IMS-LD UoL can be modelled and integrated with an external resource, i.e., an eGame created with <e-Adventure>. The ultimate goal is the full integration of external software applications and modules developed outside IMS-LD with the specification in order to improve user experience in adaptive learning while keeping in mind the need for appropriate authoring processes.
In the next sections a) we introduce a case study to motivate our approach, which will be used to illustrate the different aspects of authoring of an adaptive IMS-LD UoL supporting an external <e-Adventure> eGame; b) we describe the <e-Adventure> Project; c) we describe IMS-LD and the various options when authoring adaptive IMS-LD UoLs and when integrating externally developed modules; d) we propose a model to support the use of eGames in the adaptive learning process; and e) we explain the communication process between <e-Adventure> and IMS-LD. Earlier versions of this work can be found in [14, 15].

2. A Case Study

In order to illustrate the approach proposed, we have developed an adaptive UoL with an integrated eGame, as well as a bi-directional communication flow resulting in a personalized learning path based on two inputs: the previous knowledge and the performance of the learner [16].
In this game, called "Paniel and the Chocolate-based Sauce Adventure", the final goal is to learn more about the world of chocolate from a practical side. The student must know the properties of the ingredients and the history of this product to make exotic sauces that can suit an appropriate selection of meals and the expectations of the customers.
This eGame pursues several didactic objectives, focused on learning 1) how to make the right combination of the basic elements of chocolate to produce the base mix, 2) how to elaborate different chocolate-based sauces, and 3) how to marry a few chocolate sauces with a selection of dishes. Every objective is related to one stage of the game. The first stage (Library) deals with the origins of chocolate and the basic elaboration. The second stage (Kitchen) is more creative and allows for the elaboration of different chocolate sauces. Finally, the third part (Restaurant) is a practical exercise with customers, where the learners should obtain a perfect marriage between the dishes selected by these customers and the sauces available. The students' final grade depends on the satisfaction of the customers. The players decide on their own when they are ready to face the customers, although their level of previous knowledge influences the access point. There is no minimum number of sauces required, although there is a restriction: They need to collect and sort several ingredients and objects, and to make several sauces before being allowed to proceed to the last part (Restaurant) for the test.
The structure of the eGame and the UoL is as follows (see Figure 1): the game itself is embedded in a full UoL with a previous quiz and a post-adaptive learning path. Depending on the score obtained during the quiz, the students directly access one of the three stages of the game, where there are control questions. If the answer to one of these questions is not correct, they receive the right answer and are sent back to the previous stage. When the control question for direct access to stage 2 is right, the students automatically receive 2 mixes (dark and

milk). The students are allowed to come back to stage 1 to make more mixes at any time. When the control question for direct access to stage 3 is correct, the students automatically receive 3 sauces to be used in the game. The students are allowed to return to stage 2 to make more sauces at any time.
Once the game is over, several variables are sent back to the UoL, stating the satisfaction level of both customers and which of the possible sauces were actually prepared. The UoL takes these results regarding the learners' performance and provides an adaptive learning path out of three possible alternatives.

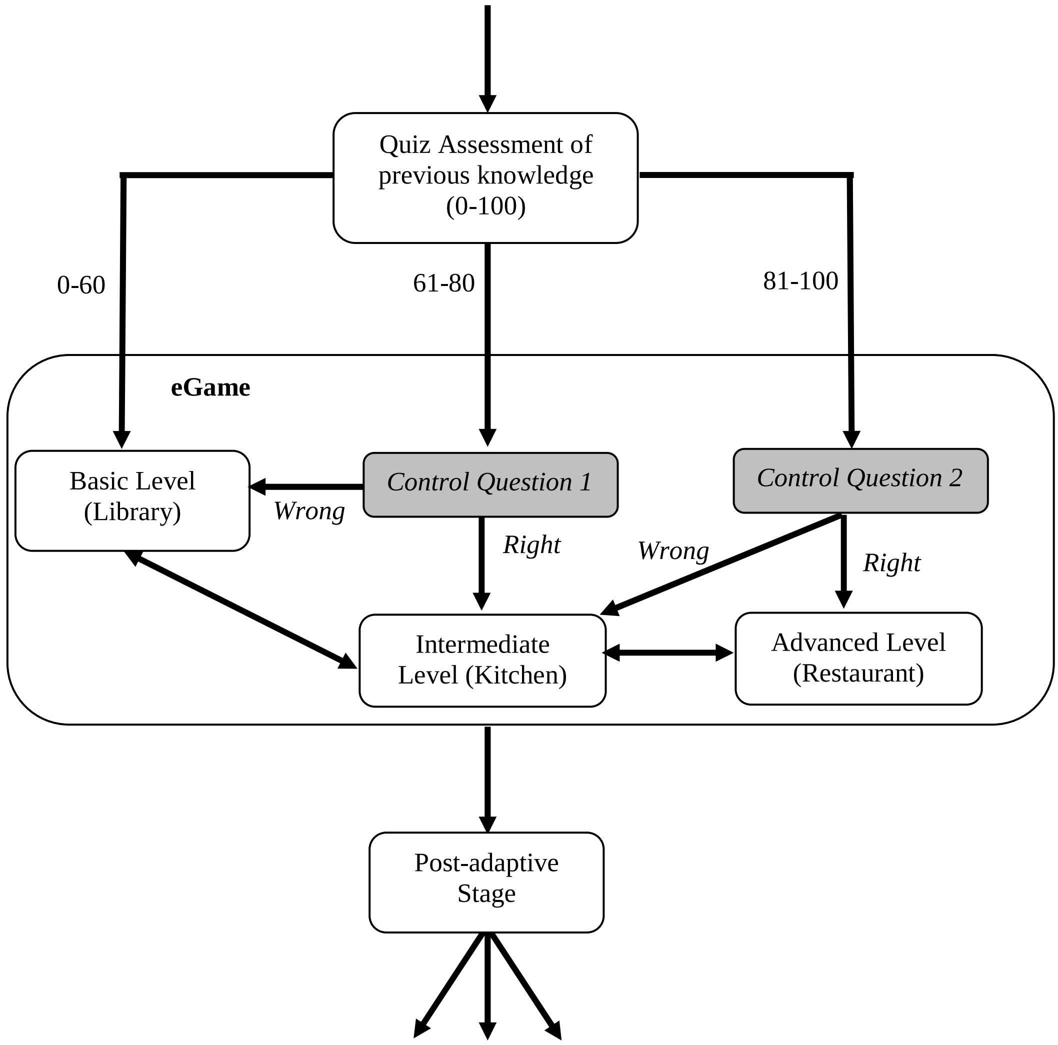

Figure 1. Structure and learning paths in the Case-Study UoL. The high-level view of the internal structure of the game is also outlined.

This case-study UoL is particularly relevant, since it integrates the typical elements arising during the production of adaptive game-based UoLs: embedded adaptive eGames, adaptive learning designs, and bi-directional communication channels (where the games affect the state of the UoL, and the UoL affects the internal state of the games). Next sections will use this case-study to illustrate the different aspects presented. This UoL is described with more detail in [15].

3. The <e-Adventure> Project

<e-Adventure> is an authoring methodology supported by an XML-based language, a graphical editor, and an engine supporting the interpretation and execution of the game. <e-Adventure> also supports personalized processes, like adaptive learning. The XML language is used to describe the environment, characters, objects and situations that form the adventure game. The objective is to let an author build and maintain an executable game without needing a previous extensive background in programming. For this purpose, the author can use a visual high-level editor that facilitates the creation of the documents that describe the games. In this section we will

give an overview of the philosophy behind the project. Further details on <e-Adventure> (formerly known as <e-Game>) are described in [17].

3.1 . The <e-Adventure> language

The first requirement of the <e-Adventure> language is simplicity. It is designed to fit in an authoring process that begins with the elaboration of the game storyboard, which is the document describing the entire game. The storyboard is an integral part of the modification, adaptation and evolution processes of the game, and thus it should be represented in a form that is descriptive and ultimately human-readable, and also easily processable by a computer. This descriptive character is one of XML’s design features [18] and the availability of mechanisms such as DTD and XML-Schema [19, 20] allows the formalization of the language in a machine-readable way. This facilitates the author’s work, by providing the means to verify the correctness of the documents before feeding them to the system.

Thus, the <e-Adventure> language is an XML-compliant markup language that closely mirrors the structure of the storyboard. Following the traits of the genre, the basic unit of construction is the “scene”. An <e-Adventure> storyboard (and thus the marked document) starts by describing all the scenes that form the game, including the associated resources, their connections to other scenes and the description of the characters and objects that populate the scene. In Figure 2 a fragment of the <e-Adventure> storyboard for the case study is sketched.

```
(...)
   <scene id="LibraryWorkshop">
      <resources>
         <asset type="background" uri="assets-chocolate/Library.jpg"/>
      </resources>

      <name>Library-Workshop</name>

      <exits>
         <exit x="750" y="0" width="50" height="400">
            <next-scene idTarget="Hall" x="100" y="0"/>
         </exit>
      </exits>

      <objects>
         <object-ref idTarget="ChocolateMixer" x="250" y="250"/>
         <object-ref idTarget="CacaoMass" x="120" y="260"/>
         <object-ref idTarget="CacaoFat" x="140" y="260"/>
         <object-ref idTarget="SoyaLecithin" x="160" y="260"/>
         <object-ref idTarget="Sugar" x="180" y="260"/>
         <object-ref idTarget="MilkForChocolate" x="200" y="260"/>
(...)
```

Figure 2. A fragment of the <e-Adventure> storyboard for the case-study. This fragment corresponds with part of the description of a game’s scene.

(a)

```
<actions>
  <use-with idTarget="ceramic_pot">
    (...)
    <effects>
      <activate flag="FINISHED_LAST_SAUCE"/>
    </effects>
  <use-with idTarget="ceramic_pot">
</actions>
```

(b)

```
<conversation id="proceed_to_final_stage">
  <condition>
    <active flag="FINISHED_LAST_SAUCE"/>
  </condition>
  (...)
```

Figure 3. Use of flags and conditions. (a) The effect of an action can be the activation of a flag. (b) A conversation with a specific character that will only be played if the condition holds.

While for further details about the <e-Adventure> language the reader is referred to [17], for the purpose of this paper it is relevant to introduce the notion of *flag*. Indeed, merely describing the elements that conform the game yields a plain structure where every door is always open, every character always says the same things, and every exit leads to the same place. For the game to be interesting, it is necessary to support the means to provide a sense of narration. We can achieve this by introducing a notion of "state". All the actions that we perform in the game should be able to affect future actions. Some objects may be hidden until something happens (e.g. the object appears only if the player has performed action X); some exits may be locked (e.g. you can't enter the library until you are admitted to the course or until you talk the secretary into letting you in); and a character may offer a different conversation (e.g. the secretary is more friendly after the player gives her a small gift).
From the perspective of the author, these interactions are conceptually modelled by allowing each interaction (with an object or character) to activate conditions or, in the <e-Adventure> terminology, *flags*. Then, the author can add preconditions to any element of the game. Intuitively, the state at any given point of the game is the set of active flags, which are an indication of which relevant actions have already been performed. Figure 3 provides a simple example of this mechanism.

3.2 . The <e-Adventure> editor

Even though the <e-Adventure> language was designed to closely resemble the structure of a storyboard, working with XML files can be a cumbersome task. For this reason, the <e-Adventure> platform also includes a visual editor that facilitates the creation of the documents that describe the games.

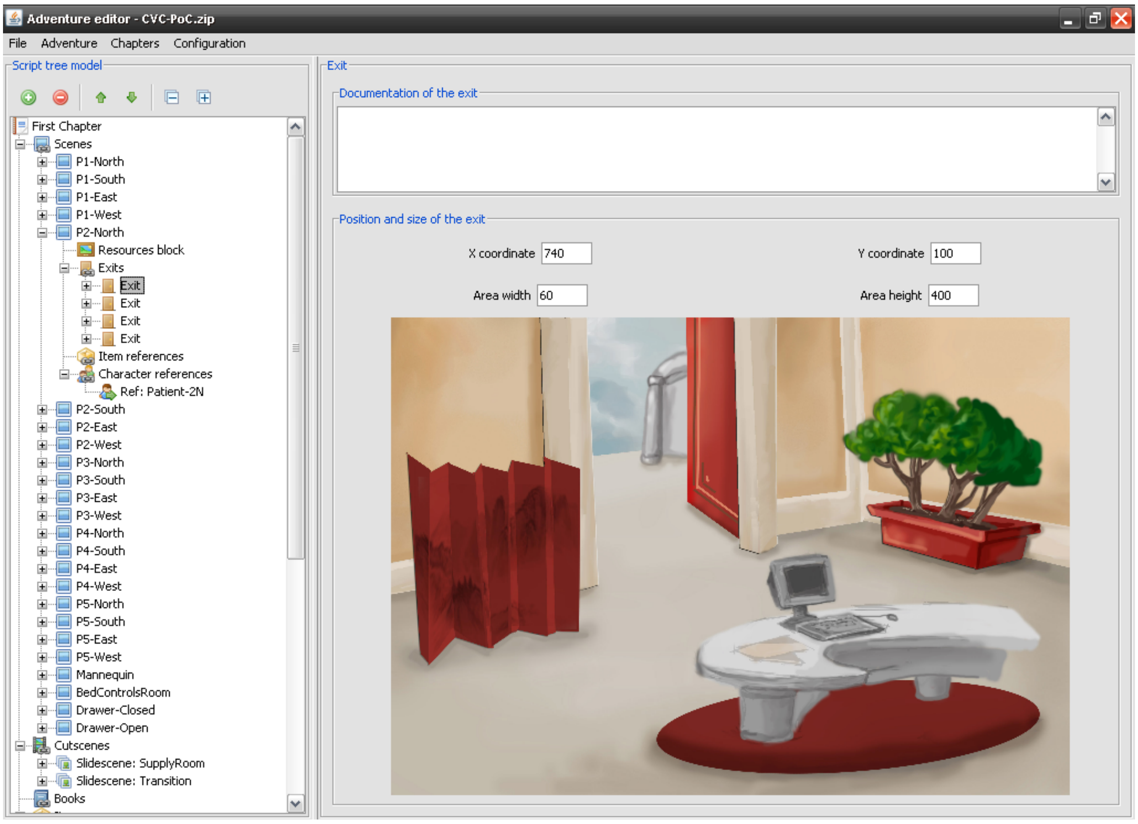

Figure 4. The <e-Adventure> visual editor. It can read and modify documents written using the <e-Adventure> language.

The editor (see Figure 4) is especially useful for dealing with coordinates or other complex aspects of creating the games, and its output are the XML files that can be interpreted by the <e-Adventure> game engine.

3.3 . The <e-Adventure> engine

The third component in <e-Adventure> is the <e-Adventure> engine. Actually, it is an interpreter for the <e-Adventure> language that takes as input a storyboard along with the different art assets required to play the game and generates the final running videogame.

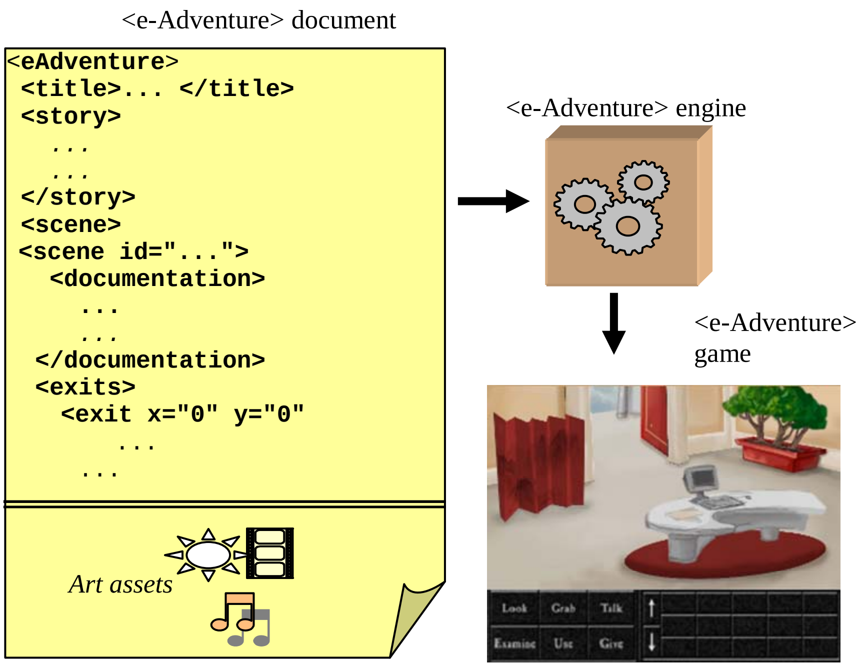

Figure 5. The <e-Adventure> engine. By processing the storyboards it can generate running videogames.

The very-high idea behind the <e-Adventure> engine is sketched in Figure 5. It is also worth noting that this engine is able to connect the game to different e-learning platforms and specifications (among them, IMS-LD compliant Learning Management Systems, as described in the next sections).

4. IMS-LD and adaptation

IMS-LD is a specification to represent and encode learning structures and methods for learners and teachers. Furthermore, IMS-LD is focused on the design of pedagogical methods able to manage learning activities linked to learning objects within a learning flow [21]. This learning flow consists of plays, acts, activities, activity structures and environments and it is flexible enough to provide several personalized itineraries depending on the role assigned or on a set of rules. Several examples are available at [22, 23]. IMS-LD consists of three levels: Level A is the main part of the specification as it provides the baseline for building any UoL with the elements Method, Plays, Acts, Roles, Role-parts, Learning activities, Support activities and Environments; Level B adds some features to create more complex lesson plans using Properties, Conditions, Calculations, Monitoring services and Global elements; and Level C adds Notifications. Each layer improves the previous one [24].
In addition to the basic structure of Level A, the elements in Level B are the actual key for more expressive UoLs (for instance, based on adaptation or collaboration), as they combine several features that make the content and the learning flow more flexible [24, 25]. Furthermore, the combination of these elements allows for modelling several classical adaptive methods (e.g., reuse of pedagogical patterns, adaptability, navigational guidance, collaborative learning, contextualized and mobile distributed learning, adaptation to stereotypes, etc), also making use of different structural elements of IMS-LD, like Environment, Content, User groups and Learning flow.
Additionally, IMS-LD is able to manage six types of adaptation [26] with a different success factor: Learning flow, content, evaluation and interactive problem solving support are fully supported. User grouping is supported by administrative tools and modification of a course on-the-fly can be partially implemented. Also interface-based modification is possible as long as the modifications are made inside the UoL and not in the player tool itself. Several of these six possibilities are also useful in managing issues complementary to adaptive learning, like active learning, collaborative learning, dynamic feedback, run-time tracking, ePortfolios and assessment [27]. In this article, we concentrate on the main type, namely the adaptation of the learning flow.
In our case-study we have used IMS-LD to outline the overall pedagogical design of the UoL. As said before, this UoL consists of three parts. The first part is focused on a quiz about the topic where the learner has to answer five multiple-choice questions to establish his/her previous knowledge. Based on this score, the learner will be granted access to a different area of the second part, the game. During the game, the performance of the learner is stored in several variables that will be sent back to the IMS-LD part of the UoL. Once the game is finished, and based on the values of these variables, the UoL will provide one learning path or another to follow.

In this case, the adaptation is present in the three parts of the UoL, based on previous knowledge and actual performance, although it could be authored to use other different types of inputs.
The first part, the quiz, is based on the UoLs Geo-Quiz 1 and 2 [22] and deals with Properties, Conditions and Calculations to define the questions and get the appropriate results out of the user's answers. It also works with adaptive content, showing and hiding different areas depending on the actual moment of the run, and providing adaptive feedback based on the user's performance. The third part, the adaptive learning path after the game, is based on Geo-Quiz 3 [22] and it allows the connection with up to three different possible itineraries as learning activities with different content (for the sake of conciseness, the actual content of these itineraries is not further detailed in this paper). All of them are hidden until the game is finished. Afterwards, only one of the activities, directly related to the performance of the user during the game is shown.
IMS-LD supports an XML binding and, in theory, any XML-compliant editor could be used to create the learning design of any UoL. However, this is not the best approach from an authoring perspective. In order to improve production and maintenance it is necessary to employ other more user-friendly authoring tools, such as those addressed in [28].

5. Types of integration of eGames in adaptive IMS-LD Units of Learning

Even though there are several experiences using eGames and simulations as part of the educational process in schools and educational environments, this use is often isolated from e-learning systems and other information packages (e.g., IMS-LD, IMS CP, SCORM) [29]. This issue stresses the disconnection between the educational setting and the authoring and execution of the contents (such as, in this case, eGames). A closer integration allows for pedagogical improvements and a better contextualized learning path [30, 31]. Also, the interoperability of the UoLs and the contents used is crucial to achieve the best educational goals in a personalized context, without compromising the rationale behind IMS-LD.

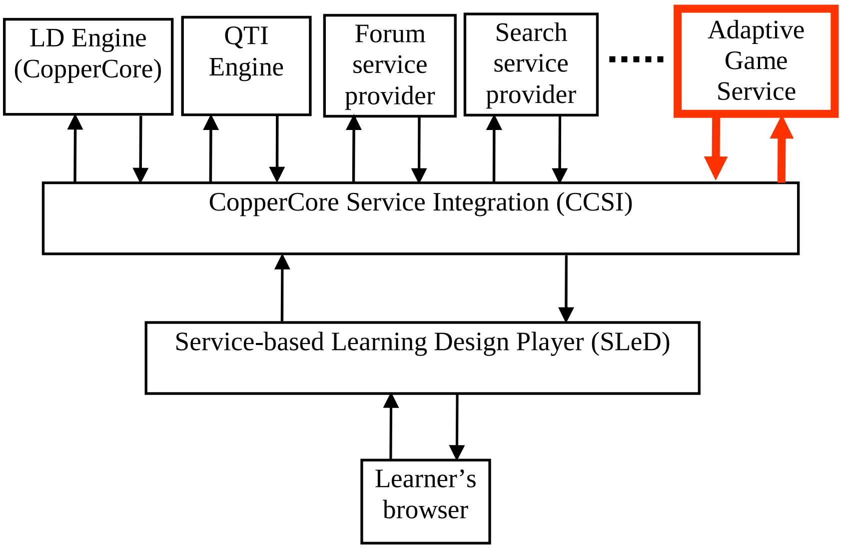

Figure 7. General architecture of communication between the learning flow and the game supported by the Coppercore / SLeD environment.

To this extent, we find two main types of integration between the UoL and eGames: a) The eGame as an embedded activity with no communication, and b) The eGame as a fully integrated resource, with bi-directional communication with the environment and a state sharing. For example, if we model a UoL containing several activities and one of them (for instance, called Activity-Game) is an eGame, the first approach will run this Activity-Game without any communication with the main flow. This means that it is impossible to establish any relations with the previous activity or the following activity inside the learning flow. Therefore, the module is incorporated into the learning flow, but remains isolated from it [30]. In the second approach, the previous Activity to the Activity-Game can provide some input to the module. For instance, the learner answers a quiz and the final score is sent directly to the eGame. Then, the eGame could start with an adaptive setting based on this input. For instance, if the score is less than a specific threshold, the starting level is for beginners; if the score is higher, the starting level is for advanced players. Last, depending on the events inside the game, a list of values is sent back to the IMS-LD learning flow to provide a detailed report and influence the next action to take

(for instance, choosing one learning path out of several possibilities). Therefore, the eGame is a fully operational part of the learning flow, able to send and receive information to and from the UoL.
This is closely related to the work carried out by Vogten et al. [32] addressing the communication between IMS-LD and IMS Question and Test Interoperability (IMS QTI) through a service integration layer named CCSI (CopperCore Service Integration Layer). This layer facilitates the communication between the IMS-LD engine (CopperCore) and the different web services required to support the different learning activities.
Following this approach, the communication between the <e-Adventure> engine and the IMS-LD Runtime Environment has been implemented by adding a new game-based adaptive service to the CopperCore / SLeD IMS-LD environment [33]. In Figure 7 the resulting high-level architecture is outlined.
This communicative behavior is particularized with the integration of <e-Adventure> and IMS-LD. Following, we describe this communication and we show how this joint approach can fulfill the integration of IMS-LD and these external resources consisting of adaptive conversational eGames.

6. The communication between <e-Adventure> and IMS-LD

As described in the previous section, the integration of eGames (implemented with <e-Adventure> or any other authoring methodology) is a complex task and raises a number of authoring issues. The basic problem is that when a learner is interacting with a specific UoL, the specification requires the runtime environment to keep a record of the state of a number of variables, called *properties*, which can be used to alter the path of the learning flow. On the other hand, eGames are often analyzed in terms of game-states, which may or may not be directly expressible in terms of IMS-LD properties.
It is necessary to provide the means to communicate and to translate the information used within the UoL and the information used within the eGames, which may require a strong programming background for the author. However, <e-Adventure> supports a clear and narrow eGame model, in which information is stored and interpreted in a declarative fashion. The task of implementing the game is facilitated by the use of a domain-specific language that can be understood and applied without a programming background. The same ideas can be applied to the authoring of the information flow, thus allowing the non-technical author to specify the communications that should take place between the eGame and the UoL in a declarative fashion. These specifications written by the author are interpreted by the adaptive game service introduced in Figure 7.
The communication between <e-Adventure> and IMS-LD is thus regulated using *adaptation* and *assessment* rules (Figure 8), which can be also be described using XML documents. This mechanism also allows authors to define the game and the UoL in parallel, because the adaptive game service can handle the transformations necessary to align (i.e., to translate) the values and variable names used in the UoL and in the <e-Adventure> game. The development process is simple and well-supported by the <e-Adventure> editor. The rest of this section describes the documents that must be created by the author to specify the mentioned rules.

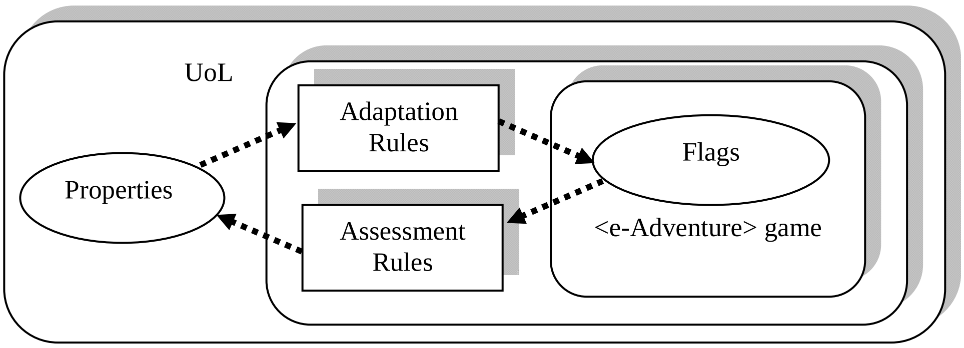

Figure 8. Communication between IMS-LD and <e-Adventure> is achieved using adaptation and assessment rules, which are interpreted by the Adaptive Game Service.

6.1 . From UoL properties to <e-Adventure> game states: Adaptation Rules

Adaptation rules close the gap that separates the internal representation of the state of a UoL and the internal representation of different initial game states, thus enabling the communication from the UoL towards the eGame. When the game is launched, the adaptive game service depicted in Figure 7 checks the state of the properties within the UoL and sees if their values trigger a specific initial game state. In case it does, the Adaptive Game Service informs the game of which flags should be activated (i.e., which is the initial state for the game).
The document with the adaptation rules identifying the relations between sets of properties and states is an XML file with the syntax exemplified in Figure 9 according to the definition of the case study. In particular, the first

part of the UoL (the initial quiz) sets two very specific properties (*go-level-2* and *go-level-3*) indicating the level of the student according to his or her responses to the quiz questions. The input configuration file for the adapter indicates which internal flags in the eGame should be activated to alter the behaviour of the game so that the simpler parts can be skipped.

```
<adaptation-rule>
  <description>Skip the first level</description>
   <uol-state>
     <property id="go-level-2" value="1"/>
     <property id="go-level-3" value="0"/>
   </uol-state>
   <game-state>
     <activate flag="finished-stage-1"/>
   </game-state>
</adaptation-rule>
<adaptation-rule>
  <description>Go to the exam</description>
  <uol-state>
    <property id="go-level-3" value="1"/>
  </uol-state>
  <game-state>
    <activate flag="finished-stage-1"/>
    <activate flag="finished-stage-2"/>
   </game-state>
</adaptation-rule>
```

Figure 9. Example of adaptation rules in <e-Adventure>. When property *go-level-2* is active, but property *go-level-3* is inactive, only the second level is stated. When *go-level-3* is active, the third level is started.

6.2 . From <e-Adventure> game states to UoL properties: Assessment Rules

Once the eGame has been designed and written using <e-Adventure>, instructors or learning designers can also prepare separate documents identifying those game states that are relevant from a pedagogical perspective and that should affect the state of the current UoL. When the <e-Adventure> engine is running, every change in the state of the game is notified to the Adaptive Game Service. The service, in turn, checks the new state against the list of pedagogically relevant states. If the state is listed, the service notifies the UoL to set the indicated properties to the corresponding values.

The document identifying the relations between states and sets of properties is an XML file with syntax that is an extension of the internal assessment engine implemented by <e-Adventure> and described in [34]. Each entry in this file is a mapping between a game state and a set of values for some of the properties present in the UoL. The game state is represented as a Boolean expression on the flags as used in the <e-Adventure> language itself (see section 3.1). Meanwhile, the properties in the UoL that should be modified are expressed with a list of set-property elements identifying the property to be set and its new value.

Given the nature of this process, it is important to note that this mechanisms supports the separation between the UoL and the definition of the game in terms of states conditioned by flags, thus allowing an authoring approach in which the writer of the game and the instructor identifying the pedagogically relevant states need not be the same person (for example, supporting a scenario in which the instructor is part of a team in which there are professional writers designing the game itself).

```
<assessment-rule id="SatCus-1-50" importance="high">
    <concept>Satisfaction Customer 1</concept>
    <condition>
       <either>
         <inactive flag="FirstCustomerSatisfiedMain"/>
         <inactive flag="FirstCustomerSatisfiedDessert"/>
       </either>
       <either>
         <active flag="FirstCustomerSatisfiedMain"/>
         <active flag="FirstCustomerSatisfiedDessert"/>
       </either>
     </condition>
     <effect>
       <set-property id="Satisfaction1" value="50"/>
     </effect>
</assessment-rule>
```

Figure 10. Example of an assessment rule described in <e-Adventure>. Depending on two flags (*FirstCustomerSatisfiedMain* and *FirstCustomerSatisfiedDessert*) the Satisfaction of Costumer 1 is set to *50*.

In our case-study the eGame keeps Boolean flags indicating which dishes with which sauces were served to each customer. Assessment rules can associate dishes delivered with punctuations that will be reported to the IMS-LD environment, thus affecting the path followed in the third part of the UoL. In Figure 10, a rule is included characterizing the first client's degree of satisfaction in terms of the mentioned in-game learner's performance.

7. Conclusions and future work

Since the beginning of IMS-LD in 2003, the relation with other surrounding technologies such as Learning Management Systems, other specifications (IMS-created or not), databases or simple stand-alone executable modules developed with any programming language, has been a pending issue. There has been a great research effort to improve the pedagogical expressiveness of the specification but not to resolve this kind of technical issues. There have been some technical initiatives aimed at achieving such integrations, like the integration between IMS-LD and IMS QTI [32] or IMS-LD and SCORM [35]. The connection between IMS-LD and <e-Adventure> and the bi-directional flow of properties that are able to modify the learning flow and the information in both sides is a contribution in this direction.

However, the introduction of this communication mechanism raises new and important issues when it comes to authoring the adaptive courses and-or UoLs. With the objective of allowing teachers and Learning Designers to model rich lectures, full of interactive learning objects, educational objectives and tasks it is necessary to provide the proper authoring tools and communication mechanism.

There are several tools and approaches to IMS-LD authoring that cover the authoring needs as far as the description of the general learning model is concerned (i.e., Reload LD Editor [36], CopperAuthor [37], Cosmos [38], LAMS [39], MOT+ [40,41], etc.). Regarding the authoring of adaptive eGames, <e-Adventure> facilitates the definition of conversational eGames for any person with a very low technical threshold. Following a simple and well-documented process, any person can design and implement small to medium-sized conversational games with special features that enhance their educational value, including adaptation and dynamic assessment.

New issues emerge when it comes to merging the two authoring processes (UoL and eGame). In this paper we have shown how the introduction of an Adaptive Game Service allows the separate development of the two elements and their posterior integration. That facilitates this task and provides very effective integration, although it can be argued that it is an *ad-hoc* solution that combines the use of separate previously-existing tools. As future work, we would like to explore the creation of combined tools that allow an integrated authoring process for the UoL and adaptive games. The process is not simple, as efficient authoring of IMS-LD designs is still a problematic task.

Acknowledgements

This paper is partially supported by the European projects TENCompetence (IST-TEL/2004-2.4.10, www.tencompetence.org) and ProLearn (IST Contract number 507310, www.prolearn-project.org), as well as the <e-UCM> research group (www.e-ucm.es). The Spanish Committee of Science and Technology (projects TIN2004-08367-C02-02, TIN2005-08788-C04-01, FIT-350100-2007-163 and TIN2007-68125-C02-01) has partially supported this work. Special thanks to Bruno Torijano Bueno for his participation in the creation and preliminary tests of the sample game for the case study.

References

[1] D. Burgos, C. Tattersall, and R. Koper, "Re-purposing existing generic games and simultations for e-learning [Special issue on Education and pedagogy with Learning objects and Learning designs]," *Computers in Human Behavior*, vol. 23, pp. 2656-2667, 2007.

[2] J. P. Gee, *What video games have to teach us about learning and literacy*. New York, USA: Palgrave Macmillian, 2003.

[3] M. Prensky, *Digital Game-based Learning*. New York, USA: McGraw-Hill, 2001.

[4] R. Garris, R. Ahlers, and J. E. Driskell, "Games, motivation, and learning: A research and practice model," *Simulation & Gaming*, vol. 33, pp. 441-467, 2002.

[5] A. Amory, "Building an educational adventure game: Theory, design and lessons," *Journal of Interactive Learning Research*, vol. 12, pp. 249–263, 2001.

[6] R. Van Eck, "Building artificially intelligent learning games," in *Games and Simulations in Online Learning: Research and Development Frameworks*, D. Gibson, C. Aldrich, and M. Prensky, Eds. Hershey, PA: Information Science Publishing, 2007.

[7] K. Squire, "Video games in education," *International Journal of Intelligent Simulations and Gaming*, vol. 2, 2002.

[8] D. Laurillard, "Multimedia and the learner's experience of narrative," *Computers in Education*, vol. 31, pp. 229-243, 1998.

[9] D. Leutner, "Guided discovery learning with computer-based simulation games: Effects of adaptive and non-adaptive instructional support," *Learning and Instruction*, vol. 3, pp. 113-132, 1993.

[10] D. Williamson, K. Squire, R. Halverson, and J. P. Gee, "Video games and the future of learning," *WCER Working paper*, vol. 2005-4, 2005.

[11] IMSLD, "IMS Learning Design," vol. 2007, Version 1 ed. Boston: The IMS Global Learning Consortium, 2003.

[12] IMSLD, "IMS Learning Design, Information Model. Objectives R2-Pedagogical flexibility and R3-Personalization," vol. 2007, Version 1 ed. Boston: The IMS Global Learning Consortium, 2003.

[13] J. L. Sierra, B. Fernández-Manjón, A. Fernández-Valmayor, and A. Navarro, "Document Oriented Development of Content-Intensive Applications," *International Journal of Software Engineering and Knowledge Engineering*, vol. 15, pp. 975-993, 2005.

[14] D. Burgos, P. Moreno-Ger, J. L. Sierra, B. Fernández-Manjón, and R. Kooper, "Authoring Game-Based Adaptive Units of Learning with IMS Learning Design and <e-Adventure>," *International Journal of Learning Technology*, vol. 3, pp. 252 - 268, 2007.

[15] P. Moreno-Ger, D. Burgos, J. L. Sierra, and B. F. Manjón, *A Game-Based Adaptive Unit of Learning with IMS Learning Design and <e-Adventure>*. Crete, Greece: Lecture Notes in Computer Science 4753 (Springer), 2007.

[16] D. Burgos, C. Tattersall, and E. J. R. Koper, "Representing adaptive and adaptable Units of Learning. How to model personalized eLearning in IMS Learning Design," in *Computers and Education: E-learning - from theory to practice*, B. Fernández Manjon, J. M. Sanchez Perez, J. A. Gómez Pulido, M. A. Vega Rodriguez, and J. Bravo, Eds. Germany: Kluwer, 2006.

[17] P. Moreno-Ger, J. L. Sierra, I. Martínez-Ortiz, and B. Fernández-Manjón, "A Documental Approach to Adventure Game Development," *Science of Computer Programming*, vol. 67, pp. 3-31, 2007.

[18] T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler, "Extensible Markup Language (XML) 1.0," in *W3C Recommendation*, vol. 2006, 2000.

[19] D. Lee and W. W. Chu, "Comparative Analysis of Six XML Schema Languages," *ACM SIGMOD Record*, vol. 29, pp. 76-87, 2000.

[20] M. Murata, D. Lee, M. Mani, and K. Kawaguchi, "Taxonomy of XML schema languages using formal language theory," *ACM Transactions on Internet Technology*, vol. 5, pp. 660-704, 2005.

[21] R. Koper and C. Tattersall, "Learning Design - A Handbook on Modelling and Delivering Networked Education and Training." Heidelberg: Springer Verlag, 2005.

[22] LN4LD, "Units of Learning developed by several authors at Learning Network for Learning Design of The Open University of The Netherlands," vol. 2006, 2005.

[23] OUNL, "DSpace," Open University of The Netherlands, 2002.

[24] R. Koper and D. Burgos, "Developing advanced units of learning using IMS Learning Design level B," *International Journal on Advanced Technology for Learning*, vol. 2, 2005.

[25] M. Specht and D. Burgos, "Implementing Adaptive Educational Methods with IMS Learning Design," presented at Adaptive Hypermedia 2006, Dublin, Ireland, 2006.

[26] D. Burgos, C. Tattersall, and R. Koper, "Representing adaptive eLearning strategies in IMS Learning Design," presented at International Workshop in Learning Networks for Lifelong Competence Development, TENCompetence Conference, Sofia, Bulgaria, 2006.

[27] D. Burgos, C. Tattersall, and R. Koper, "Utilización de estándares en el aprendizaje virtual," presented at II Jornadas Campus Virtual, Madrid, 2005.

[28] I. Martínez-Ortiz, P. Moreno-Ger, J.L. Sierra, and B. Fernández-Manjón, "Supporting the Authoring and Operationalization of Educational Modelling Languages”, *Journal of Universal Computer Science*, vol 7, pp. 938-947.

[29] P. Cobb, M. Stephan, K. McClain, and K. Gravemeijer, "Participating in classroom mathematical practices," *Journal of the Learning Sciences*, vol. 10, pp. 113–163, 2001.

[30] G. Richards, "Designing Educational Games," in *Learning Design: A Handbook on Modelling and Delivering Networked Education and Training*, R. Koper and C. Tattersall, Eds. Germany: Springer Verlag, 2005, pp. Chapter 13.

[31] D. Burgos, C. Tattersall, and E. J. R. Koper, "Can IMS Learning Design be used to model computer-based educational games?," *Binaria [www.uem.es/binaria]*, vol. 5, 2006.

[32] H. Vogten, M. H., R. Nadolski, C. Tattersall, v. Rosmalen, P., and R. Koper, "CopperCore Service Integration, Integrating IMS Learning Design and IMS Question and Test Interoperability," presented at 6th IEEE International Conference on Advanced Learning Technologies, Kerkrade, The Netherlands, 2006.

[33] OUUK, "Sled player." United Kingdom: Open University of United Kingdom, 2005.

[34] I. Martinez-Ortiz, P. Moreno-Ger, J. L. Sierra, and B. Fernández-Manjón, "Production and Deployment of Educational Videogames as Assessable Learning Objects," presented at First European Conference on Technology Enhanced Learning (ECTEL 2006), Crete, Greece, 2006.

[35] C. Tattersall, Burgos, D., Vogten, H., Martens, H., & R. Koper, "How to use IMS Learning Design and SCORM 2004 together," presented at SCORM 2006 conference, Taipei, Taiwan [http://ia.nknu.edu.tw/scorm2006/], 2005.
[36] Bolton, "Reload Project." United Kingdom: The University of Bolton, The University of Strathclyde and JISC, 2004.
[37] W. Van der Vegt, "CopperAuthor." Heerlen: Open University of The Netherlands, 2005.
[38] Y. Miao, "CoSMoS: Facilitating Learning Designers to Author Units of Learning Using IMS LD " in *Proceediongs of the International Conference on Computers in Education*, C. K. Looi, D. H. Jonassen, and I. Mitsuru, Eds. Singapore: IOS Press, 2005, pp. 275-282.
[39] J. Dalziel, "LAMS 2.0," 2006.
[40] G. Paquette, O. Marino, I. D. l. Teja, K. Lundgren-Cayrol, M. Lonard, and J. Contamines, "Implementation and Deployment of the IMS Learning Design Specification," *Canadian Journal of Learning and Technology*, vol. 31, 2005.
[41] I. d. l. Teja, K. Lundgren-Cayrol, and G. Paquette, "Transposing MISA Learning Scenarios into IMS Units of Learning," *Journal of Interactive Media in Education*, vol. 13, 2005.