



# Model and Authoring Tool to Help Teachers Adapt Serious Games to their Educational Contexts

Bertrand Marne, Jean-Marc Labat

## ► To cite this version:

Bertrand Marne, Jean-Marc Labat. Model and Authoring Tool to Help Teachers Adapt Serious Games to their Educational Contexts. International Journal of Learning Technology, 2014, 9 (2), pp.161-180. 10.1504/IJLT.2014.064491 . hal-01087301

**HAL Id: hal-01087301**

**<https://hal.sorbonne-universite.fr/hal-01087301>**

Submitted on 13 Jan 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

---

## Model and authoring tool to help teachers adapt serious games to their educational contexts

---

Bertrand Marne\* and Jean-Marc Labat

LIP6,  
Université Pierre et Marie Curie,  
4, Place Jussieu, 75005, Paris  
E-mail: [bertrand.marne@lip6.fr](mailto:bertrand.marne@lip6.fr)  
E-mail: [jean-marc.labat@lip6.fr](mailto:jean-marc.labat@lip6.fr)  
\*Corresponding author

**Abstract:** After a study, we established that one of the obstacles of the adoption of serious games (SGs) by teachers was that they cannot shape their educational scenarios to their specific teaching context. The work we present in this paper tackles the general problem of designing tools to help them customise the educational scenarios of SGs. Our approach is to provide a model suited to describe SGs that are composed of several stages, and to provide its implementation in an authoring tool in order to help the teachers to visualise, modify and check the consistency of the scenarios. The evaluation of our model shows that it is capable of describing most of the SGs we targeted, and the first user tests of our authoring tool prototype are also promising.

**Keywords:** authoring tools; adaptation; design patterns; modelling; model checking; serious games; game-based learning; teachers.

**Reference** to this paper should be made as follows: Marne, B. and Labat, J-M. (2014) 'Model and authoring tool to help teachers adapt serious games to their educational contexts', *Int. J. Learning Technology*, Vol. 9, No. 2, pp.161–180.

**Biographical notes:** Bertrand Marne is a PhD student in Computer Science at The UPMC University in Paris. He is also a Science Teacher in middle school and an ICT/TEL Tutor for teachers. As a PhD student, his research focuses on methodological and software tools to increase the acceptability of serious games for teachers.

Jean-Marc Labat is a Full Professor in Computer Science at the UPMC University in Paris. He is the Head of a research team within the LIP6 Laboratory and the Director of LUTES, a university service. He is also the current President of the ATIEF, a French TEL research association. He received his PhD from UPMC. His research focuses on cognitive modelling of users experiencing problem solving. His current research goals are to investigate serious games.

This paper is a revised and expanded version of a paper entitled 'MoPPLiq: a model for pedagogical adaptation of serious game scenarios' presented at ICALT 2013, Beijing, 15–18 July 2013.

---

## 1 Introduction

Serious games (SGs) for educational purposes have become more numerous and more diverse over the past years. They now target all levels of education and cover a large variety of domains and subjects (Squire, 2005). However, their adoption by teachers remains outside the mainstream and much lower than one would expect (De Grove et al., 2012; Azadegan et al., 2012). Indeed, several identified obstacles that prevent the adoption of SGs have been identified. For some of them, there are scientific studies aiming to solve these problems. For instance, some authors highlight the cost and complexity of the design and development of SGs and provide some authoring tools (Moreno-Ger et al., 2005; Marfisi-Schottman et al., 2010; Göbel et al., 2008) and some methodologies (Yusoff, 2010; Kiili, 2005; Marne et al., 2012b) to deal with these issues. One of the typical problems that limits the adoption of SGs by teachers is their inappropriateness with the educational context. Indeed, after interviewing teachers, we identified that one of the obstacles that hinders the use of SGs in their classrooms is the fact that their pedagogical scenarios could not be adapted to suit their specific teaching context. Moreover, the adaptability of tools (especially information and communication technology – ICT) is a crucial point for their adoption by teachers: SGs should not remain static learning objects. There is a need for tools that empower teachers and allow them to adapt SGs according to their needs and contexts of use (Egenfeldt-Nielsen, 2004).

The work presented in this paper tries to meet the general objective of designing tools to help teachers adapt the educational scenarios of SGs. More precisely, we present two aspects of our approach: on one hand, the design of a model and a visualisation capable of describing SG scenarios in a way that allows teachers to restructure them pedagogically. On the other hand, the design of an authoring tool prototype to help teachers to visualise scenarios and meant to help them to maintain the scenarios' consistency and the gameplay even after they have made many meaningful pedagogical adaptations, thanks to a dedicated consistency validation tool.

Before detailing these two aspects of our approach, we will present in the next section how the needs expressed by teachers led us to our research questions. Then, in Section 3, we will present the specific points we chose in the related scientific work to base our own work. This will lead us to present, in Sections 4 and 5, the visual language called MoPPLiq and the authoring tool called APPLiq, we are providing to help teachers customise SGs to their needs. Finally, before concluding, we will discuss the results of various assessments we carried out on MoPPLiq.

## 2 Research questions

It has been shown that when ICT tools are not flexible and cannot be adapted by teachers, there is little chance for them to be adopted (Ouraiba et al., 2010). As explained below, a small study we conducted with teachers shows that this is also the case for SGs. This study was made in two phases. In the first phase, we conducted several brainstorming sessions, focus groups and interviews. In the second one, we used questionnaires and interviews.

In the first stage of this study, we met teachers and worked with them on the pedagogical scenario of an SG currently being developed (Donjons & Radon) (<http://www.ad-invaders.com/project.php?id=19>). They were asked to design a course on

the three states of water (solid, liquid and gas) and the transitions between these states. To our surprise, none of them had designed similar courses. Even though the learning objectives were the same (i.e., those in school curricula), the order in which they were taught was completely different. After questioning the teachers, we understood that the different choices they had made were based on the diversity of their educational contexts and were dependent on the tools available in their schools, their teaching habits and the means of collaboration with teachers of other disciplines, etc. Even though we had made this observation, the resources of the project only allowed us to develop one of these scenarios and, as we had predicted, two thirds of the teachers were not satisfied with this final version and would have preferred sequencing the educational objectives of the SG in a different way. The only way to provide these teachers with an SG that suits all of them is, therefore, to provide an SG with an alterable pedagogical scenario: i.e., a scenario in which they can change the sequence of the learning objectives.

Because the teachers we interviewed on Donjons & Radon were ICT experts and in order to confirm these conclusions with ‘mainstream’ teachers, we carried the study on to a second stage. We submitted a questionnaire to several teachers who were candidates for the use of the SGs Refraction (<http://play.centerforgamescience.org/refraction/site/>) and CellCraft (<http://www.cellcraftgame.com/>). Among the questions, we asked ‘what were the shortcomings of these SGs?’, and ‘what could we do to improve them?’ In most cases, teachers complained about the activities in the SGs (number, sequence, etc.). And, they proposed modifications such as changing the order of the learning objectives. E.g., ‘there should be a menu to choose which level to play’ (CellCraft), ‘only the worlds 3, 5 and 7 are useful for my students’, ‘some levels are too hard. We should be able to avoid them’ (Refraction), etc.

These observations led us to work on the adaptation of SG scenarios, especially SGs that can be split into distinct components (any kind of scenario with a sequence of levels, stages, exercises, case studies, etc.). Our main goal is to enable teachers to customise the SGs scenarios to their needs. We split this problem into several research questions we will address in the following sections. Which model is suitable for representing an SG scenario in order to facilitate its adaptation? Which software implementation is the best to help teachers adapt SGs to fit their needs? How can we help the teachers modify an SG scenario while still maintaining a coherent storyline?

Before we discuss these issues, in the next section we will analyse related work and extract the core features we have used as foundations for our work.

### **3 Related work**

In order to build our model for restructuring scenarios of SGs according to the needs of teachers, we have reviewed a large selection of the scientific studies on authoring tools and their underlying models. We first focused on scenario models related to the field of SGs. But as research in this field is still new and scarce, we have broadened the scope of our review to the field of technology enhanced learning (TEL) and video games.

Thus, through the study of each of these three fields, we managed to identify three basic features that characterise scenario models. In the following subsections, we will present each of these features.

### 3.1 *Goal oriented approach*

The first feature we extracted in all of the three fields studied is the division of the scenarios into components defined by goals. This feature is present in most educational modelling languages (EML) such as IMS-LD (Koper and Olivier, 2004).

For example, LAMS (Dalziel, 2008) is an authoring tool for intelligent tutoring systems (ITS) intended for teachers. The authors claim that the model is a simplification of IMS-LD. Indeed, their model breaks down scenarios into several components defined with educational goals. There are other ways to break down a scenario. In the field of TEL systems, ScenEdit (Emin et al., 2010) is an authoring tool designed to assist teachers in the formalisation of pedagogical scenarios. The components in ScenEdit's model (i.e., ISiS) are defined with the teachers' intentions. This same type of breaking down method is also used in the SG field. For instance, defining components of the scenarios with teachers' intentions was partially used by ScenLRPG (Mariais et al., 2012). This authoring tool for learning role playing games (LRPG) provides a model in which components are defined not only with intentions, but also with interactions and educational goals. Legadee (Marfisi-Schottman, 2012; Marfisi-Schottman et al., 2010) is another interesting authoring tool meant to help design 'learning games'. In Legadee, the scenario model is based on IMS-LD (Koper and Olivier, 2004), which facilitates the fragmentation into components but it also uses the metaphor of the cinema: the scenario is divided into shots, which have actors, objects, and scenery. WEEV (Marchiori, 2010) is another model and a tool to conceptualise scenarios before implementing them in the SG authoring tool, eAdventure. WEEV splits scenarios on a geographical basis: each component is a place. Otherwise, in the field of video games, the division of scenarios into several levels is also fairly standard. Moreover, this division is based mostly on the nature of the challenges featured (Aponte et al., 2011). Björk and Holopainen (2005) made a design pattern called 'Levels' which summarises the concept of breaking down game scenarios into several components: "A level is a part of the game in which all player actions take place until a certain goal has been reached or an end condition has been fulfilled".

### 3.2 *Hierarchy of goals vs. hierarchy of stages*

Regarding the idea of dividing scenarios into levels or stages, we wanted to highlight another feature that is often found in scenario models described in scientific papers: their components (e.g., levels) are organised and connected by the hierarchical structure of their goals. For instance, SG authoring tools like ScenLRPG (Mariais et al., 2012), or LAMS (Dalziel, 2008) and Collage (Hernández-León et al., 2006) for TEL systems, provide a hierarchical structure of components with their models. At first glance, their components look as if they are only sequenced in a linear fashion, but the components can also be nested: some components are containers of other sequences of components. With LAMS, these nested components are used for conditional branching (Dalziel, 2008) and are based on the hierarchy of educational goals. Other TEL systems also allow nesting [e.g., AHA! (Bra and Calvi, 1998) or ScenEdit (Emin et al., 2010)], and the same applies in the field of SGs [e.g., WEEV (Marchiori, 2010) or Storytec (Göbel et al., 2008)], and even in the field of video games (Kearney and Pivec, 2007; Aponte et al., 2011). A good example of nested components and branched scenarios based on the hierarchy of educational goals is Legadee's model (Marfisi-Schottman et al., 2010;

Marfisi-Schottman, 2012). Indeed, its hierarchy is based on a movie metaphor (shots, sequences, chapters) on the gaming side and on IMS-LD on the educational side.

As in the previous subsection, we have unified these concepts and managed to extract their guiding principle thanks to design patterns. Indeed, design patterns are by nature syntheses of best practices aiming to describe the best solution to common problems. The design pattern for SGs (Marne et al., 2012a, 2012b) ‘Pedagogical hierarchy of goals’ is derived from the ‘Hierarchy of goals’ from Björk and Holopainen (2005) and summarises the issue of the hierarchy of components based on goals: “How to make the scenario linking educational goals consistent while remaining playful and taking into account performance and player choices?” The main idea here, is first to build a hierarchy of educational goals (i.e., knowledge, skills, attitudes), and secondly to build nests to construct hierarchically the connections between levels.

### *3.3 Dynamic adaptation of the components*

The last feature of the models of authoring tools that we have chosen to highlight in the perspective of an adjustment of SG scenarios, is the fact that scenario components can dynamically adapt to the choices and performances of the ‘serious-player’. In some TEL authoring tools, the issue of adapting the storylines to the learners’ profiles can be simply implemented by showing and hiding some components. The implementation can be more complex while detailing, summarising and illustrating elements (Murray, 2003) depending on the learners’ profiles. In more complex cases, TEL systems adapt the storyline to the learner’s profile by modifying the behaviour of the components themselves. For instance, in adaptive hypermedia (Bra and Calvi, 1998; Brusilovsky, 1996), in each page (i.e., component) some links are shown or hidden depending on the user’s profile. Once again, several design patterns for video games deal with this dynamic adaptation. Björk and Holopainen (2005) provide two design patterns to help the designers with adaptive components/levels: ‘Supporting goals’ and ‘Optional goals’. Their main idea is to offer secondary objectives, often optional, that help players in trouble, or rather, spice up challenges for the most experienced ones.

We chose to use a bottom-up research method to build our model for restructuring scenarios of SGs. Therefore, our model partly relies on the experience of our research team and our private partners in the field of SG design, but it mostly relies on the three aspects of the related work we introduced above: breaking down scenarios into components defined by their goals, modelling non-linear scenarios to anticipate the serious players’ actions and their consequences, and being able to describe the dynamic adaptations to the serious players model, when it is available. In the next section, we will present how we used these three aspects to build the foundations of our model.

## **4 MoPPLiq, a model for SGs scenario**

In the previous section, we presented the three fundamental features that we identified in previous research on models of scenarios. We have chosen to build our own model on these features in order to help teachers to customise SG scenarios. Our model is called MoPPLiq (stands for ‘Modélisation des Parcours Pédago-Ludiques’ in French and means: model for gaming aspects and educational aspects of SG scenarios.)

In the following subsections, we will describe MoPPLiq showing how it integrates all three aspects.

#### 4.1 *Breaking down scenarios into ‘black boxes’*

Some SGs are composed of a succession of components with a unique game-play (e.g., Refraction). In this case, it is very easy to model the scenario: we simply model one component and then vary its settings to represent all the other components of the SG. For the other SGs, which are made of components that have various game-plays (e.g., the quests and mini-games of Game for Science)<sup>1</sup>, it would not be possible to design a generic model of the components’ inner mechanisms. Moreover, the components’ specific gameplay is not important when it comes to adapting the scenario. What really matters is its impact on the evolution of the scenario. In other words, we need to know the impact that each component has on the serious-player’s model and the game settings, especially if they have an impact on the other components. This is why we decided to model the components of the SG scenarios as ‘black boxes’ called ‘activities’. To describe the activities in MoPPLiq, we chose to characterise them by the educational and recreational goals they help achieve and that have an impact on the serious-player’s model and the SG’s scenarios.

**Figure 1** Part of the XML file describing level 6.2 of refraction (see online version for colours)

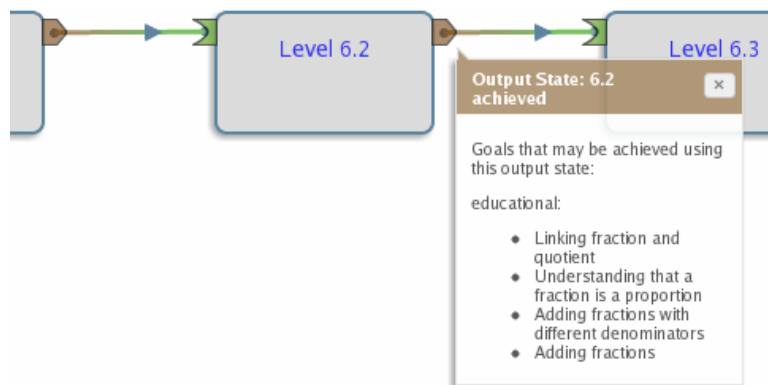
```
<activity id_activity="11">
  <name>Level 6.2 </name>
  <description>Tutorial to understand that you can "add" lasers with the same
denominator.</description>
[...]
  <input_state id_input="9" />
[...]
  <output_state id_output="13"/>
[...]
</activity>
<goals>
  <goal id_goal="7" type="ludo">
    <name>You can bend the beam using the laser bender</name>
    <goal_links>
      <goal_link object="output_state" id_object="13" />
[...]
    </goal_links>
  </goal>
[...]
  <goal id_goal="24" type="edu">
    <name>Understanding that a fraction is a proportion</name>
    <goal_links>
      <goal_link object="output_state" id_object="13" />
    </goal_links>
  </goal>
[...]
  <goal id_goal="38" type="edu">
    <name>Adding fractions with different denominators</name>
    <goal_links>
      <goal_link object="output_state" id_object="13" />
    </goal_links>
  </goal>
[...]
</goals>
```

For example, refraction, an SG intended to help students learn to manipulate fractions (divisions, multiplications, additions), is divided into activities that meet educational and recreational goals. Level 6.2, for example, has the following educational goals: ‘understanding that a fraction is a proportion’, ‘adding fractions with different denominators’, etc. The MoPPLiq formalism describes level 6.2 as shown in Figure 1.

We use the element `<activity>` to describe the activity itself. With the `<goals>` elements that reference the output (`<output_state>`) of the activity, we also characterise several of its goals (educational goals have an ‘edu’ type attribute, and recreational goals have a ‘ludo’ type attribute). The `<goal>` elements can also be used in reference to the `<input_state>` (i.e., the beginning of the activity). Thereby we use them to set prerequisite goals that are necessary to enter an activity. Please note that `<goal>` elements describe the objectives that can be achieved within a player’s game and are not intended to describe external goals such as externally acquired competences or other players’ data.

Linear scenarios, like the one in refraction, are constructed by linking the activities with their inputs and outputs. Figure 2 is the graphical representation of Figure 1.

**Figure 2** Level 6.2 of refraction, modelled by an activity (rounded box) characterised by goals (see online version for colours)



Note: Educational goals are described in the information bubble.

#### 4.2 Non-linear scenarios

SGs often have non-linear scenarios, where the links between activities are conditioned by the serious-player’s actions which can sometimes be related to the acquired knowledge of the player. For instance, in Les ECSPER (<http://campus-douai.gemtech.fr/course/view.php?id=934>) (that stands for ‘Études de Cas Scientifiques et Pratiques pour l’Expertise en Rupture’ in French and means: scientific and practical case studies for the fractures analysis), the choices made by the serious-player in each activity sets the nature of the next activity.

For example, in one of the activities of this SG, the serious-players must infer whether the failure mode of a screw is brittle or ductile. If the answer is wrong, the next activity will be a support activity in which the learners will be given a short course on the subject so that they understand their mistake. If the answer is correct, the next activity



will allow learners to conduct further examinations of the screw, and the educational sub-goal ‘recognise a ductile failure mode’ will be considered to have been ‘worked on’.<sup>2</sup>

Figure 3 shows how MoPPLiq is able to describe this kind of situation. The activity contains several `<output_state>` elements that correspond to a serious-player’s choice and that are linked to different activities (links are described with the `<output_input_link>` element). The wrong answer (i.e., ‘Brittle’) leads to the support activity and the right answer (i.e., ‘Ductile’) leads to the next examination activity and it is linked to the educational sub-goal that was ‘worked on’ (`<goal_link object="output_state" id_object="34"/>`). As you can see, this sub-goal is also a prerequisite for the activity ‘Examination of the surface of the fracture’.

**Figure 3** Part of the XML file describing Les ECSPER’s non-linear scenario (see online version for colours)

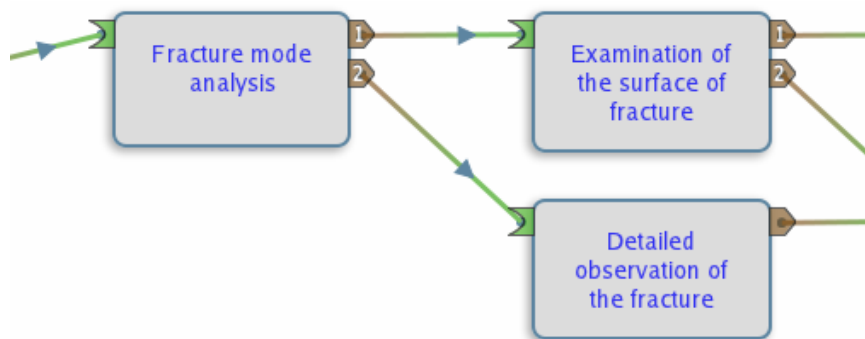
```
<activity id_activity="24">
  <name>Fracture mode analysis</name>
  [...]
  <output_states>
    <output_state id_output="34" name="Ductile" />
    <output_state id_output="35" name="Brittle" />
  </output_states>
</activity>
<activity id_activity="25">
  <name>Examination of the surface of the fracture</name>
  [...]
  <input_state id_input="40" />
  [...]
</activity>
<activity id_activity="30">
  <name>Detailed observation of the fracture</name>
  [...]
  <input_state id_input="32" />
  [...]
</activity>
<links>
  <output_input_link id_output="34" id_input="40" />
  <output_input_link id_output="35" id_input="32" />
</links>
<goals>
  <goal id_goal="23" type="edu">
    <name>Recognize a ductile failure mode</name>
    <goal_links>
      <goal_link object="output_state" id_object="34" />
      <goal_link object="input_state" id_object="40" />
    </goal_links>
  </goal>
</goals>
```

Unlike the previous example (Figures 1 and 2), for purposes of the present explanation, we chose a simplified example that has only one sub-goal ‘worked on’. But, keep in mind that an output state does not represent the purpose of the activity itself. However, it represents a choice of the serious-players that is indexed by the subset of goals

(‘sub-goals’) that can be ‘worked on’ by them if they made that particular choice. As we pointed out in the previous section, these sub-goals are intended to describe only the actions of the current player in the game and their consequences. These sub-goals cannot describe external states such as the situation of other players or availability of some external resources. This limitation was necessary to introduce a scenario consistency validation system.

By allowing activities to have several output states with MoPPLiq, we are able to model non-linear scenarios that adapt to the serious-player’s choices and performances. The graphical representation of such a scenario describes the activities linked with a tree structure (see Figure 4).

**Figure 4** Example of a non-linear scenario (see online version for colours)



Note: Brown tips marked 1 and 2 are the output states that represent the players’ choices and lead to different activities.

### 4.3 Adaptable activities

Advanced SGs often adapt their behaviour to the serious-player’s model. To illustrate this behaviour, let us take the example of Game for Science, a massively multiplayer role-playing SG. In this SG, there is a quest called ‘Water you waiting for?’ that teaches the various techniques to reduce the pollution of a river. Its first activity is a lab analysis exercise that has two operating modes. The first mode is the ‘beginner-mode’ that offers a tutorial to the serious-player. The second mode is the ‘experienced-mode’: without any help, the serious-player must perform a correct lab analysis. Depending on the serious-player’s model (i.e., having performed a lab analysis before, or not), the activity does not have the same behaviour. To express these different modes and their connection to the serious-players’ model (i.e., the goals they worked on in previous activities) with our model MoPPLiq, we use several *<input\_state>* elements for each activity as you can see in Figure 5.

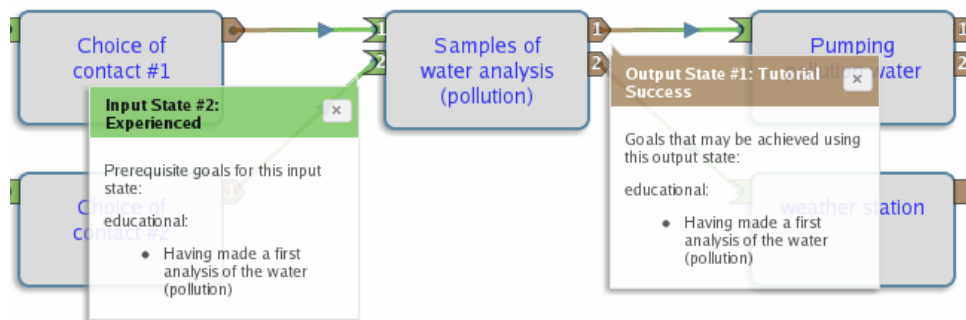
The *<goal>* ‘Having made the lab analysis tutorial’ is linked to the ‘Experienced’ *<input\_state>* as a prerequisite. It is also linked to the ‘Beginner’ *<output\_state>* as a ‘worked on’ educational objective. For example, a novice learner that starts the activity with the ‘Beginner’ input state exits with the ‘Beginner’ output state and his serious-player model is updated with the goal ‘having made the lab analysis tutorial’. Figure 6 and Figure 8 show a visual representation of this part of the model. As we explained in the previous work review, we were very much influenced by EML principles and especially

by IMS-LD's while designing our model. Indeed, input states and their related goals can be respectively understood as simplified conditions and properties of IMS-LD Level 2 (Koper and Olivier, 2004). To stay easily readable and understandable by teachers, and to maintain a model that can be easily searched for inconsistencies, we simplified the 'conditions' concept. Thus, we use only one type of condition (a sort of 'if present') to apply on one type of properties: the prerequisite and 'worked on' goals.

**Figure 5** Part of the XML file showing several input states used to model distinct behaviours of an activity of the SG Game for Science (see online version for colours)

```
<activity id_activity="15">
  <name>Samples of water analysis (pollution)</name>
  <input_states>
    <input_state id_input="17" name="Beginner" />
    <input_state id_input="18" name="Experienced" />
  </input_states>
  <output_states>
    <output_state id_output="25" name="Successful Tutorial" />
    <output_state id_output="26" name="Success on your own" />
  </output_states>
</activity>
<goals>
  <goal id_goal="41" type="edu">
    <name>Having made the lab analysis tutorial</name>
    <goal_links>
      <goal_link object="output_state" id_object="25" />
      <goal_link object="input_state" id_object="18" />
    </goal_links>
  </goal>
</goals>
```

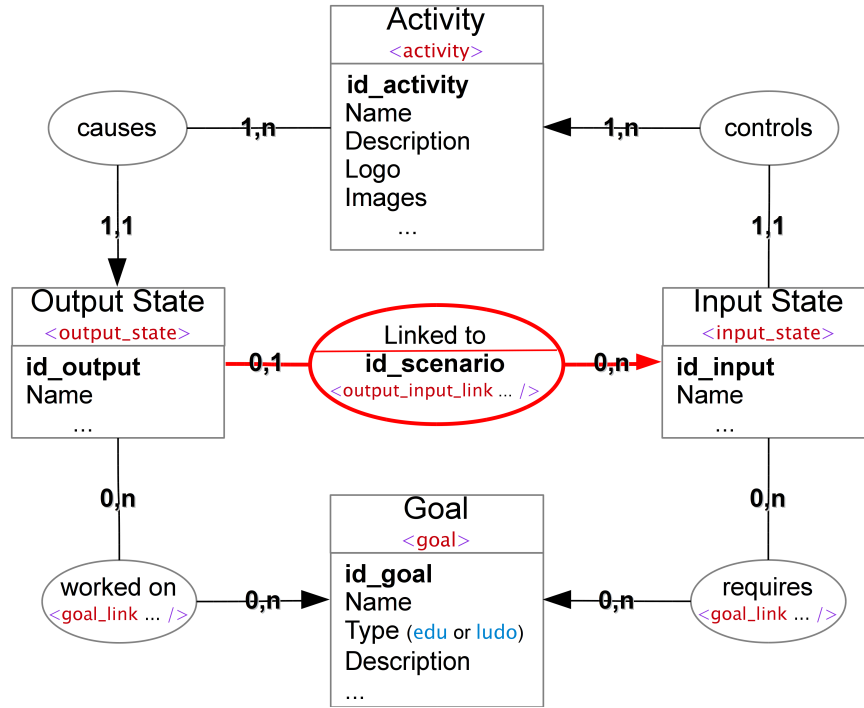
**Figure 6** Example of an activity able to change its behaviours depending on the serious-player model (see online version for colours)



Notes: Green chevrons marked 1 and 2 are the input state that represents the activity's possible behaviours. Each input state has its own prerequisites.

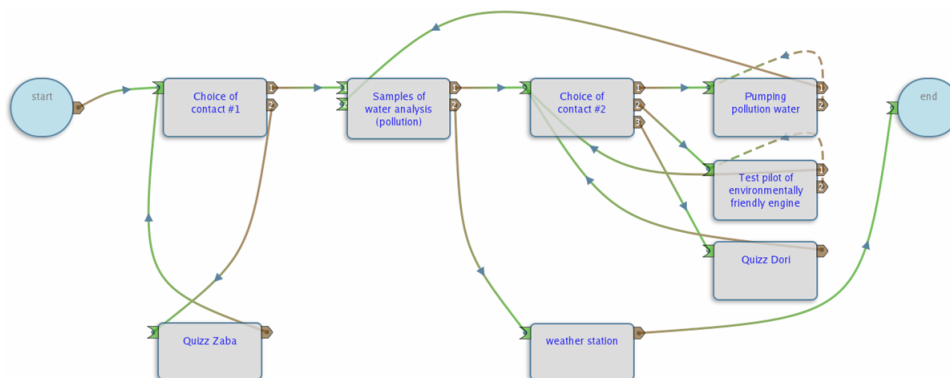
The whole model formalism (activities, goals, input and output states and relationships between them) that we presented with XML examples can also be presented with an entity-relationship (ER) diagram (Figure 7).

**Figure 7** ER diagram of the MoPPLiq model (see online version for colours)



To provide teachers with a visual representation of the scenario flow graphs (e.g., Figures 2, 4, 6 and 8) and a set of tools to modify it, we implemented the MoPPLiq authoring tool. This authoring tool also features a model checking system to help the teachers maintain the logic in the storyline that we will describe in the next section.

**Figure 8** Graphical representation of the scenario of the quest ‘Water you waiting for?’ (Game for Science) (see online version for colours)



## 5 APPLiq: authoring and validation tool

Our authoring tool prototype which implements the model MoPPLiq and helps teachers adapt SG scenarios is called APPLiq (APPLiq stands for ‘Adaptation des Parcours Pédago-Ludiques’ in French and means: adaptation of the scenarios blending gaming and educational aspects). This tool has two different purposes. Of course, its first goal is to help teachers adapt pedagogical SG scenarios to their context. Its secondary objective is to help designers model their SG using MoPPLiq so that teachers can then customise them.

We chose to make APPLiq as a web application in order to allow teachers and designers to share the scenarios they create and modify easily. Therefore, APPLiq was developed using PHP and MySQL for data manipulations, and also uses HTML/CSS and JavaScript for the graphical user interface (GUI).

In the following subsections, we will describe how we designed APPLiq centred on the teachers’ main interest: adapting the pedagogical scenario of SGs.

### 5.1 *Graphical visualisation and GUI intended for teachers*

To help teachers understand the SG scenarios and provide them with tools to manipulate them, we followed two paths. On one hand, APPLiq provides a graphical visualisation of the activity flow. On the other hand, APPLiq provides a GUI that highlights the pedagogical aspects of the storyline because they are the teachers’ main concern.

Thus, APPLiq represents activities (see Figure 8) with entry points, symbolised by chevrons, and exit points, placed on the right represented by tips. The activity flow goes from left to right. This visualisation is directly inspired by Virtools and 3DVIA Studio which are design tools for 3D simulation, and SGs. We were also inspired by the graphical representations used in authoring tools for video games (e.g., Unity, Kismet, Cry Engine Sandbox, etc.)

In order to help teachers stay focused on their goals and needs (i.e., creation and modification of the educational scenario), we chose to show as much educational information as possible in the GUI and also to hide most of the gaming information.

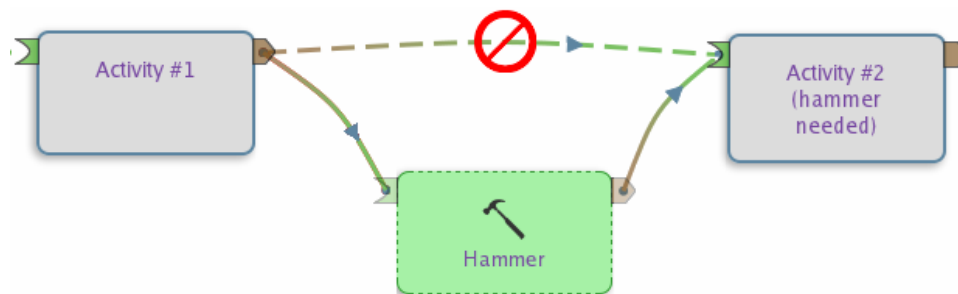
### 5.2 *Maintain logic of the storyline*

In order to help the teachers adapt their scenarios without creating contradictions in the SG storyline, we designed a system in our authoring tool that detects and solves inconsistencies that may appear during the modifications. Thus, when an educational prerequisite goal cannot be matched in the scenario, creating a pedagogical inconsistency, APPLiq raises an alert so that the teachers can rearrange or add activities accordingly. But these educational changes are not mandatory, because even if an educational scenario of an SG may seem inconsistent on its own, this may not be the case in its context of use (i.e., other training materials, initial knowledge of the serious-players, etc.). In addition, if the modifications brought to the scenario create inconsistencies in the storyline, APPLiq offers the possibility of adding ‘buffer activities’.

Let us explain this last concept with a basic example: suppose a teacher has structured a scenario with an activity that has the prerequisite ‘the serious player must carry a

hammer' because it will be used. However, in the scenario the teacher has designed, none of the previous activities allow the serious-player to obtain a hammer. To solve this inconsistency, the authoring tool automatically offers to insert a buffer activity that allows the serious-player to get the hammer before the activity that requires it (Figure 9). To avoid disrupting the educational choices made by the teacher, these buffer activities have no educational purposes or prerequisite conditions.

**Figure 9** Example of buffer activity insertion in our authoring tool (see online version for colours)



## 6 Checking MoPPLiq and testing APPLiq

To check the expressivity of MoPPLiq, we chose two paths. On one hand, we tried to model many SGs with MoPPLiq and even video games to check if we were able to describe them. On the other hand, we worked on importing functionalities capable of importing data from other SG authoring tools into MoPPLiq, in order to check if our model was able to formalise all the elements available in the underlying scenario models of these authoring tools.

First, we will present the above assessment work carried on MoPPLiq in the following subsections. Then, we will present our initial user tests with APPLiq's prototype.

### 6.1 Modelling SGs with MoPPLiq

To assess MoPPLiq, we tried to model different kinds of SGs with it. We first chose accessible SGs that we can play online, and then, with the help of interns, we started to model each of them using APPLiq. The 17 SGs (see Table 1) we chose are mostly freely available online, and most of them have a scenario that can be easily split into components (e.g., levels, exercises, case studies, etc.).

In most cases (10 out of 17), these SGs have a linear scenario which fits very well with MoPPLiq (see Refraction on Figure 2). Except for one of them, we were able to describe these SGs fully. We only had problems with CellCraft because we were not able to describe the overall storyline, which introduces every level, with gaming goals ('worked on' and prerequisites). Thus, the game is correctly described with MoPPLiq, but we cannot manage inconsistencies with APPLiq during storyline adaptation.

**Table 1** Results of modelling SGs with MoPPLiq

<i>Serious games</i>	<i>Types of scenario</i>		<i>Dyn. adapt.</i>	<i>Time</i>		<i>Loops</i>	<i>Dec. goals</i>	<i>Global events</i>
	<i>Linear</i>	<i>Branched</i>		<i>Seq.</i>	<i>Cont.</i>			
Academy Island	OK	-	-	OK	-	-	-	-
Ayiti	-	NOK	-	-	NOK	NOK	NOK	NOK
CellCraft	OK	-	-	OK	-	-	-	-
Défenses Imm.	-	OK	-	OK	-	-	-	-
Donjons & Radon	OK	-	-	OK	-	-	-	-
Game for Science	-	OK	OK	OK	-	OK	NOK	-
Facteur Academy	-	NOK	-	OK	-	OK	-	-
FoodForce	OK	-	-	OK	-	OK	-	-
Harmoniculteurs	OK	-	-	OK	-	-	NOK	-
Les ESCPER	-	OK	-	OK	-	-	-	-
Ludiville	OK	-	-	OK	-	-	-	-
McDonald's video game	-	NOK	OK	-	NOK	OK	-	NOK
Mecagenius	-	OK	OK	OK	-	-	-	-
Mon coach APB	OK	-	-	OK	-	OK	-	-
Prog & Play	OK	-	-	OK	-	-	-	-
Refraction	OK	-	-	OK	-	-	-	-
Starbank	OK	-	-	OK	-	-	-	-

Notes: 'Dyn. adapt.' means dynamic adaptation to the serious-player model (see also the Section 4.3). 'Seq.' stands for sequential time and means that the time is not global. 'Cont.' stands for continuous time and means that time is counted globally. 'Dec. goals' stands for decrementing 'goals' and means the possibility of decrementing indicators. 'OK' means that we were able to model this aspect of the SG. 'NOK' means that we were unable to model this aspect with MoPPLiq. 'Défenses Imm.' stands for the SG 'Défenses Immunitaires'.

Seven out of 17 SGs tested have a non-linear scenario (with branches). Thanks to the output states (e.g., Les ECSPER in Figure 4), we were able to model most of them with MoPPLiq correctly. We had serious problems with two of them. The first one is Ayiti, and our main problem was that this SG cannot be easily broken down into components such as levels. But, Ayiti has also several features that are not currently implemented in MoPPLiq. Thus, in this SG, the flow of time is not sequenced with activities or levels: some independent events can be triggered off (e.g., a timer ends, a season passes, etc.) and then interrupt the course of the ongoing activity, branching the serious-player to another one. We encountered the same issue with McDonald's video game (events in the restaurant). The global event feature is not yet implemented in MoPPLiq, but we are working on it. This implementation raises several problems with inconsistency detection in APPLiq. There is also another feature of Ayiti that we have trouble to integrate into MoPPLiq. This feature is the decrementing 'goals' (Dec. goals). This awkward name describes the problem of modelling indicators that may decrease during the SG. In Ayiti, the family embodied by the serious-player can earn money, this is easily modelled by MoPPLiq through the acquisition of 'worked on' goals when crossing the output states. However, the serious-player can also lose money (the indicator decreases), which we currently cannot model well. At first, we thought of introducing scores or percentages on the goals ('worked on', and prerequisites). But we realised that for some SGs with a



scenario containing loops (e.g., 6 out of the 17 SGs we have modelled), it becomes hazardous to check their consistency in APPLiq (because we cannot know beforehand how many loops the serious-player will play).

Except for McDonald's video game that we were not able to model (see above), the input states were very useful to describe the dynamic adaptation (Dyn. adapt.) of SG scenario's components to the serious-player profile. It was the case for only two SGs out of the 17 we modelled: Game for Science (see Figure 8) and Mecagenius.

To conclude, we were able to model fully or partially 14 out of the 17 SGs we tried (see Table 1). Our main problem was to manage continuous elements such as global storyline or timers in MoPPLiq and APPLiq knowing that we made these tools for discrete components. We also had some difficulties with describing decrementing indicators with our binary goal system. Therefore, this evaluation of the expressivity of MoPPLiq shows that it is suitable to model most of the SGs tested.

Although this kind of assessment, based on the modelling of several SGs, cannot be thorough, the positive results are nevertheless very encouraging for us. To complete and confirm these results, we undertook another type of assessment of MoPPLiq based on model transformation.

## 6.2 Importing data from other SG authoring tools

To cover a broader spectrum of SG types, we tried to import data directly from other SG authoring tools into APPLiq. The goal was to measure what part of the models implemented into these authoring tools we are able to express with the MoPPLiq's formalism. We chose to use two very different authoring tools to test a wide range of concepts with our model: we tried to import files from  Legadee (Marfisi-Schottman et al., 2010; Marfisi-Schottman, 2012) and  eAdventure (Moreno-Ger et al., 2006).

### 6.2.1 importing from Legadee

We were very successful importing Legadee's files. Indeed, its underlying model is not far from ours and we were able to express most of the elements defined in Legadee's model with MoPPLiq (see Table 2). There is only one feature that we were not able to express: the fact that the activities are nested in bigger groups of activities. This feature can be useful to help teachers organise and visualise the activities of complex scenarios and we are therefore considering the relevance of integrating it into our model.

### 6.2.2 importing from eAdventure







Importing eAdventure files was a lot more challenging because its underlying model is really different from MoPPLiq's. The main difficulty was to identify the links between activities because they are referenced in several types of elements such as objects, conversations, cut-scenes, etc. Nevertheless, we succeeded in expressing most of eAdventure's elements with our model (see Table 2). However, we ran into the same problem that we had while modelling the SG Ayiti: the only element that we did not manage to transpose into MoPPLiq is the centralised system that handles events and that is capable of interrupting an activity and starting another one. For instance, Fire Protocol (an SG designed with eAdventure) features a timer that triggers off events at specific



moments in the game and we were not able to formalise it in MoPPLiq (Figure 10 shows the Fire Protocol scenario modelled with MoPPLiq).

The results of model transformations from other SG authoring tools into APPLiq confirm the results obtained from the previous SG modelling assessment: the expressivity of MoPPLiq is good for describing most of the elements of SGs that can be split into distinct stages even though there are still some improvements to be made such as adding the possibility of modelling nested components. Moreover, even if it seems difficult to introduce continuous elements in this fundamentally discrete model, we are working on adapting MoPPLiq so that it could describe global events and stories.

**Table 2** Comparing formalisms of Legadee and eAdventure with MoPPLiq (see online version for colours)

MoPPLiq elements	 Legadee	 eAdventure
 Activities <activity>	Activities correspond to the <screen> element	Activities basically correspond to the <scene> elements but they can also correspond to some cutscenes (<slideshow>, <videoscene>, <graph-conversation>) and conversation elements when they are able to trigger another scene, cutscene or conversation.
 Input States <input_state>	Input states correspond to the ending points of the <Connector> elements.  (For a same <screen> element, input states are distinguished with the <condition> nested in <Connector> elements)	Input states correspond to the targets of <next-scene> (nested in <exit>) and the <trigger-xxxx> elements (where xxxx can be related to the type of 'scene', such as scene, slideshow, videoscene or conversation).  (For a same target element, input states are distinguished when target associated attributes such as <b>DestinyX</b> , <b>Y</b> or <condition> elements are different).
 Output States <output_state>	Output states correspond to the starting points of the <Connector> elements.	Output states correspond to each <exit> elements or <trigger-xxxx> elements nested in a <scene> pointing to a distinct target (targets are distinguished with the <condition> and <effect> elements, and also with the <b>DestinyX</b> and <b>Y</b> attributes).
 Goals <goal>	'Worked on' goals correspond to the <competency>, <knowledge> and <behaviors> elements that are linked to each <scene>.  'Prerequisite' goals correspond to the <condition> elements of the incoming <Connector> elements.	Goals correspond to the <condition> flags used to link activities.  'Worked on' goals correspond to the flags nested in <effects> elements.  'Prerequisite' goals correspond to the flags nested in <condition> elements.

### 6.3 Initial testing for APPLiq

Our approach is currently limited by a major problem: there are no SGs on the market able to import a modified scenario. Thus, if we can model games with APPLiq, we cannot export the modified or adapted scenarios so that they can be executed by the SGs. In a joint Ph.D. with a private partner (KTM Advance), our team has developed a framework for SG development called Genome. This framework is based on the paradigm ‘entity systems’ (or CES for component entity system) (Gestwicki, 2012), which greatly helps the development of adaptable SGs, since objects of the SG are fully described with XML in each component. Thus, with the help of Genome we are currently working on developing several adaptable SGs.

Pending the completion of the development of these SGs, we only tested the modelling features of APPLiq, but our first results are encouraging. Indeed, 3 out of 3 interns and 14 out of 14 computer science students who tried to model SGs with MoPPLiq preferred doing it through the APPLiq’s GUI than directly with XML (even though they are used to writing XML code).

Thanks to a preliminary study, we showed that teachers feel the need to adapt SGs to their needs and their educational context, by modifying the educational scenario. To address this problem, we have chosen to answer three research questions: Which model is suitable for the representation of an SG scenario to facilitate its adaptation? Which software implementation is the best to help teachers adapt SGs to fit their needs? How can we help the teachers modify an SG scenario while still maintaining a coherent storyline?

To answer the first research question, our approach has been to base our research and the resulting model on previous work, and in particular on design patterns. Thus, we based our model called MoPPLiq on an innovative combination of three features:

Breaking down SG scenarios into several activities ('black boxes') whose outputs and inputs are indexed with recreational and educational sub-goals ('worked on', or prerequisites). Modelling the choices and performances of the serious-player in an activity thanks to various output states connected to the input states of the following activities. Using different input states to distinguish the behaviours of an activity depending on the serious-player model. Input states and output states can both be indexed with either educational and recreational goals.

We tested the expressivity of this model using two methods. On the one hand, we attempted to model about 20 SGs with MoPPLiq and, on the other hand, we tried to import data from other SG authoring tools by carrying out a model transformation. The results of these tests indicate that MoPPLiq is sufficiently expressive to describe adequately SGs with scenarios divided into distinct stages (e.g., levels, exercises, case studies, etc.). These tests also show the limits of our model: the fact that MoPPLiq breaks the scenarios down into discrete components currently renders it very difficult to describe continuous elements (i.e., timers, events or stories out of sync with activities). In addition, our model currently does not handle nested components. However, we are working on ways to address these two shortcomings in the future versions of MoPPLiq.

APPLiq is the authoring tool that implements MoPPLiq and aims to answer our two other research questions. Thus, it allows a new type of graphical visualisation of SGs modelled with MoPPLiq that is meant for teachers' understanding, and provides a GUI centred on educational aspects, meant to help them to customise the scenarios. Moreover, APPLiq embeds an innovative system for detecting and compensating inconsistencies. Based on the system of 'buffer activities', its purpose is to enable teachers to handle scenarios without worrying about creating inconsistencies (especially on the gameplay).

Although APPLiq is still a prototype, we have begun testing it with users playing the role of SG designers who try to model their products. The promising results encourage us to go further in the development of this tool and also to develop more flexible SGs that can be customised by the teachers with the help of MoPPLiq and APPLiq.

## Acknowledgements

The authors would like to thank the MOCAH team and especially Dr. Iza Marfisi and Dr. John Wisdom for their very helpful support. In addition, we thank interns for their contribution to our research: Hanene Bouguerra, Romain Gugert and Clément Rouanet. This project is partially supported by a grant from the Région Île de France, that the authors wish to thank.

## References

- Aponte, M-V., Levieux, G. and Natkin, S. (2011) 'Measuring the level of difficulty in single player video games', *Entertainment Computing*, Vol. 2, No. 4, pp.205–213.
- Azadegan, A., Riedel, J.C.K.H. and Hauge, J.B. (2012) 'Serious games adoption in corporate training', in Ma, M., Oliveira, M.F., Hauge, J.B., Duin, H. and Thoben, K-D. (Eds.): *Serious Games Development and Applications, Lecture Notes in Computer Science*, pp.74–85, Springer, Berlin, Heidelberg.
- Björk, S. and Holopainen, J. (2005) *Patterns in Game Design*, 1st ed., Charles River Media, Hingham, Mass., USA.

- Bra, P.D. and Calvi, L. (1998) 'AHA! An open adaptive hypermedia architecture', *New Review of Hypermedia and Multimedia*, Vol. 4, No. 1, pp.115–139.
- Brusilovsky, P. (1996) 'Methods and techniques of adaptive hypermedia', *User Modeling and User-Adapted Interaction*, Vol. 6, No. 2, pp.87–129.
- Dalziel, J. (2008) 'Using LAMS Version 2 for a game-based learning design', *Journal of Interactive Media in Education*, No. 2 [online] <http://www-jime.open.ac.uk/jime/article/view/2008-24> (accessed 26 July 2014).
- De Grove, F., Bourgonjon, J. and Van Looy, J. (2012) 'Digital games in the classroom? A contextual approach to teachers' adoption intention of digital games in formal education', *Computers in Human Behavior*, Vol. 28, No. 6, pp.2023–2033.
- Egenfeldt-Nielsen, S. (2004) 'Practical barriers in using educational computer games', *On the Horizon*, Vol. 12, No. 1, pp.18–21.
- Emin, V., Pernin, J-P. and Aguirre, J.L. (2010) 'ScenEdit: an intention-oriented authoring environment to design learning scenarios', in Wolpers, M., Kirschner, P.A., Scheffel, M., Lindstaedt, S. and Dimitrova, V. (Eds.): *Sustaining TEL: From Innovation to Learning and Practice, Lecture Notes in Computer Science*, pp.626–631, Springer, Berlin, Heidelberg.
- Gestwicki, P. (2012) 'The entity system architecture and its application in an undergraduate game development studio', in *Proceedings of the International Conference on the Foundations of Digital Games, FDG '12*, ACM, New York, NY, USA, pp.73–80.
- Göbel, S., Salvatore, L. and Konrad, R. (2008) 'StoryTec: a digital storytelling platform for the authoring and experiencing of interactive and non-linear stories', in *2008 International Conference on Automated Solutions for Cross Media Content and Multi-Channel Distribution*, Florence, Italy, pp.103–110.
- Hernández-Leo, D., Villasclaras-Fernández, E., Jorrín-Abellán, I., Asensio-Pérez, J., Dimitriadis, Y., Ruiz-Requies, I. and Rubia-Avi, B. (2006) 'Collage, a collaborative learning design editor based on patterns', *Educational Technology & Society*, Vol. 9, No. 1, pp.58–71.
- Kearney, P.R. and Pivec, M. (2007) 'Recursive loops of game-based learning: a conceptual model', in *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2007*, AACE, Vancouver, Canada, pp.2546–2553.
- Kiili, K. (2005) 'Digital game-based learning: towards an experiential gaming model', *The Internet and Higher Education*, Vol. 8, No. 1, pp.13–24.
- Koper, R. and Olivier, B. (2004) 'Representing the learning design of units of learning', *Educational Technology & Society*, Vol. 7, No. 3, pp.97–111.
- Marchiori, E.J. (2010) *WEEV: A Multidisciplinary Approach to Educational Game Development*, Master Thesis, Madrid, Spain, Complutense University at Madrid [online] <http://eprints.ucm.es/11349/> (accessed 26 July 2014).
- Marfisi-Schottman, I. (2012) *Méthodologie, modèles et outils pour la conception de Learning Games*, Thèse de Doctorat en Informatique, Lyon, France, Université Claude Bernard Lyon 1.
- Marfisi-Schottman, I., George, S. and Frank, T-B. (2010) 'Tools and methods for efficiently designing serious games', in *Proceedings of ECGBL 2010 The 4th European Conference on Games Based Learning*, Danish School of Education Aarhus University, Copenhagen, Denmark, pp.226–234.
- Mariais, C., Michau, F. and Pernin, J-P. (2012) 'A description grid to support the design of learning role-play games', *Simulation & Gaming*, Vol. 43, No. 1, pp.23–33.
- Marne, B., Wisdom, J., Huynh-Kim-Bang, B. and Labat, J-M. (2012a) 'A design pattern library for mutual understanding and cooperation in serious game design', in *Proceedings of the 11th International Conference on Intelligent Tutoring Systems (ITS 2012), Lecture Notes in Computer Science*, Springer, Berlin/Heidelberg, pp.135–140.

- Marne, B., Wisdom, J., Huynh-Kim-Bang, B. and Labat, J-M. (2012b) 'The six facets of serious game design: a methodology enhanced by our design pattern library', in Ravenscroft, A., Lindstaedt, S., Kloos, C. and Hernández-Leo, D. (Eds.): *21st Century Learning for 21st Century Skills, Lecture Notes in Computer Science*, Springer, Berlin/Heidelberg, Saarbrücken, Germany, pp.208–221.
- Moreno-Ger, P., Martínez-Ortiz, I. and Fernández-Manjón, B. (2005) 'The <e-Game> project: facilitating the development of educational adventure games', in *Proceedings of the IADIS International Conference on Cognition and Exploratory Learning in Digital Age (CELDA 2005)*, Porto, Portugal.
- Moreno-Ger, P., Martínez-Ortiz, I., Sierra, J.L. and Fernández-Manjón, B. (2006) 'Language-driven development of videogames: the <e-Game> experience', in *Entertainment Computing – ICEC 2006: Proceedings (Lecture Notes in Computer Science ... Applications, Incl. Internet/Web, and HCI)*, Springer, Cambridge, UK.
- Murray, T. (2003) 'An overview of intelligent tutoring system authoring tools: updated analysis of the state of the art', in *Authoring Tools for Advanced Technology Learning Environments*, pp.491–544, Kluwer Academic Publishers, Cop., Pays-Bas, Dordrecht; Boston; London.
- Ouraiba, E.a., Choquet, C., Cottier, P., Després, C. and Jacoboni, P. (2010) 'Engineering of open learning scenarios – the case of hop3x learning scenarios', in *2010 IEEE 10th International Conference on Advanced Learning Technologies (ICALT)*, pp.264–268.
- Squire, K. (2005) 'Changing the game: what happens when video games enter the classroom', *Innovate: Journal of Online Education*, Vol. 1, No. 6, pp.1–20.
- Yusoff, A. (2010) *A Conceptual Framework for Serious Games and its Validation*, Thesis, Southampton, University of Southampton.

## Notes

- 1 <http://www.gameforscience.com/> (see also Section 4.3).
- 2 The lack of a sophisticated tracking system renders it impossible to know whether a sub-goal such as 'recognise a ductile failure mode' is actually reached or not and this is why we chose to use the term 'worked on' instead of achieved.