HAL
open science

A Model-Driven Engineering Process to Support the Adaptive Generation of Learning Game Scenarios

Pierre Laforcade, Esteban Loiseau, Rim Kacem

To cite this version:

Pierre Laforcade, Esteban Loiseau, Rim Kacem. A Model-Driven Engineering Process to Support the Adaptive Generation of Learning Game Scenarios. CSEDU, INSTICC, Mar 2018, Funchal, Madeira, Portugal. pp.67-77. hal-01740062

HAL Id: hal-01740062

https://hal.science/hal-01740062v1

Submitted on 20 Feb 2024

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.

A Model-Driven Engineering process to support the adaptive generation of learning game scenarios

Pierre Laforcade[1], Esteban Loiseau[1], Rim Kacem[1]
[1]*Computer Laboratory of Le Mans University, France*
{*pierre.laforcade, esteban.loiseau*}*@univ-lemans.fr*

Keywords: Serious Games, Adaptation, Learning Game Scenario, Generation, Metamodeling.

Abstract: Adaptation is a key-concern when developing serious games for learning purposes, particularly for people with specific needs. This paper presents a Model-Driven Engineering framework that eases the design and the validation of adaptive learning scenarios. It tackles the personalization issue by helping domain experts and computer scientists in making explicit then in validating the domain elements and rules involved in the adaptation. The framework proposes a metamodeling process based on a metamodel specifying at first the domain elements according to both a 3-incremental-perspective on the resulting scenario, and a 3-dimensions specification of domain elements to use and produce. The framework then proposes to model the game description and the child's profile as input models for the generator that will produce the adapted scenario as an output model. We applied the framework in the context of the *Escape it!* project that aims at helping young children with Autistic Syndrome Disorder to learn and generalize visual performance skills.

1 INTRODUCTION

The use of serious games (Deterding et al., 2011) in Autistic Syndrome Disorder (ASD) interventions has become increasingly popular during the last decade (Ern, 2014). They are considered as effective new methods in the treatment of ASD. Computerized interventions for individuals with autism may be much more successful if motivation can be improved and learning can be personalized by leveraging principles from the emerging field of serious game design in educational research (Whyte et al., 2015).

This paper tackles the challenge of adapting learning sessions to the needs of individual learners. Our research is conducted in the context of the *Escape it!* project. The objective is to develop a serious game to train visual skills of children with ASD. This serious game will borrow mechanics from "escape-room" games: the player's goal is to open a locked door to escape the room. To this end, the user has to solve numerous puzzles often requiring observation and deduction. We adapted it for our targeted audience, requiring to solve only one puzzle per scene.

In the context of this project we are focusing at first on the generation of learning scenarios adapted to the current learning progress of children with ASD. Indeed, this generation is difficult because there are a lot of elements and rules involved in the set up of an adapted game session. We need an approach that allows experts to participate in making explicit and formalizing the domain elements and rules involved during the generation of an adapted learning scenario. We propose such a framework: it is based on Model-Driven-Engineering (MDE) principles and tools. MDE (Mussbacher et al., 2014) is a research domain promoting an active use of 'executable' models throughout the software development process, leading to an automated generation of the final application. We already tackled instructional design challenges with MDE techniques in the past (Loiseau et al., 2015) however the learning scenarios were specified by teachers and not generated by the machine.

The remainder of this paper is as follows. The next section presents the *Escape it!* project contextualizing our research. Section 3 discuses a literature review about the generation of adapted scenarios. Our proposition is then presented in Section 4. Finally, a case-study is presented in Section 5 in order to apply and validate our proposition in the context of the project. We discuss in the conclusion our results and draw the next directions for further research.

2 THE *ESCAPE IT!* PROJECT

This section presents the *Escape it!* project that contextualizes our research.

2.1 Objectives of the project

The project aims at designing and developing a mobile *learning game* (a serious game with learning purposes) dedicated to children with ASD (Autistic Syndrome Disorder). The game intends to support the learning of visual skills (based on the ones described in the ABLLS-R® curriculum guide (Partington and Analysts, 2010)). The mobility feature will allow the learning to take place wherever the parents, therapeutics, as well as the child himself want it. The game will be used both to reinforce and generalize the learning skills that will be initiated by "classic" working sessions with tangible objects.

The project involves autism experts, parents and Computer Science researchers, experts in the engineering of Technology-Enhanced Learning systems.

2.2 General overview and principles for the serious game

The serious game is based on the *escape-the-room* structure: players have to find hidden cues and objects, sometimes combine objects or interact with the playing scenes, in order to unlock a door. The door symbolizes the end of the level and gives access to the next level. We made a cross analysis between mechanisms from various escape-the-room-*like* games and best practices for designing serious games for children with ASD (Ern, 2014)(Zakari et al., 2014).

We then sketched with experts, during participatory design sessions, the major directions and requirements to take into account. Some of them are related to the game aesthetics and sound environment (to be adapted to the child sensory profile), other about the regulation of the child activity (prompts, guidances, feedbacks, reinforcements, etc. have to be adapted to every children profile), or about the tracking system that will be used to update the children profiles after a game session. We only focus here on the scenario involving the resolution activities, and list, below, the most appropriate informations for our concern:

- targeted skills: a subset from the 27 visual performance skills depicted in the ABLLS-R, the relevant ones for a mobile gameplay adaptation.
- variable game sessions: a session will propose from 3 to 6 levels according to a start menu choice (to the convenience of the pairing adult with the child or the child him-self).
- levels as meaningful living places: the whole screen will display a fixed (no scrolling or change of point of view) and enclosed (with a door to open) easily identifiable living places. These scenes are related to themes, for example the *bedroom, kitchen and living room* are related to the *home* theme, whereas *classroom* and *gymnasium* are related to the *school* theme.
- adapted difficulty: according to the current child progress in the targeted skills; difficulty raises after three succeeded activities for a same skill (along one or several game sessions).
- generalizing the acquired skills: to this end, the scenes have to change, in accordance with the previous difficulty level, in order to propose non-identical challenges for the same skill; i.e. :
  - changing the scene (background and elements);
  - adding background elements to disrupt visual reading;
  - changing the objects to find and handle;
  - adding other objects not useful to the resolution;
  - hiding objects behind or into others.

We developed several prototypes to test our hypothesis and try the first designs. Figures 1, 2, 3 and 4 depict the 4 levels of a scenario taking place in the same scene (*gymnasium* from the *School* theme). These levels intend to support the learning of 4 visual skills: match object to picture, match object to object, sort into categories, and seriation.

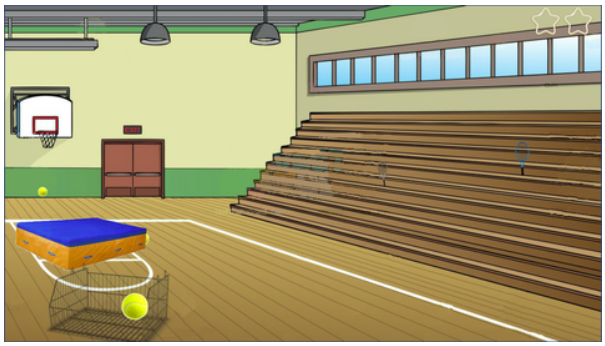

Figure 1: Resolving the B3 visual skill (matching an object to an identical object). In this example two tennis balls have to be found and moved into the metal storage.

2.3 Anatomy of a scene

Whichever scenes are selected for the learning scenario, they share common constructs and rules:

- a background image that depicts a familiar scene for the children, with a few recognizable objects;

Figure 2: Resolving the B4 visual skill (matching an object to an identical picture): a tennis ball is found and moved into the metal storage. The previously closed door is now open to gain access to the next level.

Figure 3: Resolving the B8 visual skill (sorting several objects into different categories): 4 objects to be found and moved into the 2 storage boxes according to their categories.

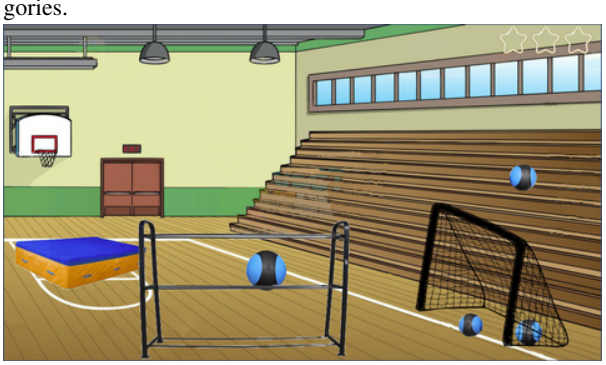

Figure 4: Resolving the B25 visual skill (completing a serialization). In this example, a ball is already positioned on the ball storage; 3 others balls of different sizes have to be found and positioned.

- several empty slots where objects to find can be later positioned;
- additional decorations to impair visual reading (in relation to the difficulty rules); each one can:
  - appear in different locations;
  - create new slots for other game objects;
- interactive *hiding objects*: provide new hidden objects slots and reveal them when touched;
- solution objects where game objects have to be dropped in/on;
  - one or several places can be proposed to position this element or the different instances required to solve the level (e.g. for sorting objects two or more storage boxes can be used);
  - zero or several places can be proposed if dropped objects have to appear specifically.

Prompts, guidances, feedbacks will also affect the visual aspect of a scene (e.g. the empty/fulled stars in Figures 1 to 4) but are not relevant to the scenario we are focusing on.

The current prototype implements a 4-levels scenario taking place in the *gymnasium*. The proposed scenes deal with 4 visual performance skills. Except the solutions objects and the range of choices for the relevant objects to handle with the targeted skill, all objects, additional decorations, positions, hiding elements, etc. are randomly selected and positioned. This prototype does not currently handle the child's profile. Its purpose was to experiment and randomly display combinations of playable scenes.

2.4 Design issues and research problematics

As stated above, a game scenario is composed of an ordered sequence of scenes including precise descriptions of each scene components and locations. All this information has to be adapted to the child's profile when starting a new game session. There are various profile variables (current progress during learning skills, preferences/dislikes, level difficulties for every skill, etc.) and a lot of combinations of elements to set-up a scene. In addition, generalization is very important in autism. It can be defined as the process of taking a skill learned in one setting and applying it in other settings or different ways (Leaf and McEachin, 1999). To support the learning of generalization, we have to propose a large variety of scenes settings.

Nevertheless, it is not possible to design and develop all the combinations of settings. We need dynamically generated game sessions, adapted to each child's profile. Next section studies existent solutions.

3 Literature review

The adaptation of Technology-Enhanced Learning systems allows the personalization of learning by adapting whole or parts of the presented systems (contents, resources, activities, etc.) to the learner's needs, interests and abilities. An adaptable learning

system is generally considered as the degree in which the system can be manipulated by its end-users. Differently, adaptive learning environments aim at supporting learners in acquiring knowledge and skills in a particular learning domain; they can adapt to the needs of the learner. In our context, our first focus is not on the adaptability of the mobile learning game for children with ASD but on the adaptivity of the system: to provide adapted game sessions in accordance to the children' profiles.

Several studies tackled the adaptation issue in serious games. For example, (Lopes and Bidarra, 2011) noticed that adapting to specific skills is more important than to the global notion of difficulty or challenge like in video games or simulations. The learning goals to achieve are usually strongly coupled with the gradual personal improvement of a skill set. Adaptive serious games generally have specialized, and *ad hoc*, approaches, where game components are adjusted to encourage training of a specific skill. However adaptivity differs on the targets (game mechanics, AI, narratives, content, etc.) and the methods (bayesian networks, ontologies, neuronal networks, rules-based systems, procedural algorithms, etc.) (Sina et al., 2014)(Janssens et al., 2014)(Callies et al., 2015).

(Sehaba and Hussaan, 2013) proposes a generic architecture for personalizing a serious game scenario according to learners' competencies and interaction traces. The architecture has been evaluated in the context of the CLES project with the objective to develop a serious game for evaluating and rehabilitating cognitive disorders. It is organized in three layers: domain concepts, pedagogical resources and serious game resources. They focus on the techniques (machine learning) to update the learner profile using interaction traces (Hussaan and Sehaba, 2016). It does not match our current adaptation challenge, nevertheless their 3-layers architecture and the process to generate the learning scenarios are close to our concerns. They also propose the generation of 3 successive scenarios (conceptual, pedagogical and serious game scenarios) according to the 3-layers. They used an evaluation protocol to validate the generated scenarios. Experts were involved at first to validate the domain rules, *a priori* of the generator implementation, and then to produce scenarios, for specific contexts, that will be compared to the generated ones. Experts are considered as requirements and validation sources but are not directly involved in the specification process.

Differently in the EmoTED project, a serious game for helping children with ASD to train about the mimic of emotions, we proposed a model-driven approach to more involve experts in the design and validation of game mechanisms (Laforcade and Vakhrina, 2015). These researches did not tackle the generation issue but the model-driven principles and process could be further studied for our current challenge.

4 A model-driven framework for adaptive generator

4.1 Overview of the framework

The rationale of our proposition is to combine the general architecture of a scenario generator from the CLES project (Sehaba and Hussaan, 2013) with the Model-Driven Engineering approach and process used in the EmoTED project (Laforcade and Vakhrina, 2015) to support the specification of the elements (concepts, properties and rules) required to drive the adaptive generation of scenarios. From the first architecture we follow the generator principle where the final learning game scenario is built after 3 steps. Similarly we propose to split the final scenario generation into three scenarios (named differently) :

- the **objective scenario** refers to the selection of the targeted learning objectives according to the user's profile including his current progress. In the *Escape it!* context it is related to the elicitation of the visual performance skills in accordance to the number of levels to generate.
- the **structural scenario** refers to the selection of learning game exercises or large game components. For our case-study: the various scenes where game levels will take place. This scenario extends the previous one (i.e. the objective scenario elements are included in this one). It is generated from user profile elements and knowledge domain rules stating the relations between these pedagogical large-grained resources and the targeted skills they can deal with.
- the **features scenario** refers to the additional selection of the inner-resources/fine-grained elements. In the *Escape it!* project it concerns all objects appearing in a scene. The game scenario includes both previous scenarios components; it specifies the overall information required by a game engine to drive the set-up of a learning game session.

It is worth noting that the final scenario is only composed of required descriptive informations to adapt the game sessions to a user's profile. Informations about how these descriptions elements are functioning (static and dynamical dimensions) might be ad-

dressed by the generator and the serious game engine: they are out the scope of the scenarization process.

From the second MDE-oriented research work, we follow the metamodeling/modeling approach and the use of the EMF framework (*Eclipe Modeling Framework*)(Steinberg et al., 2009). We propose to capture the global domain elements, required for the generation, into three inter-related parts of a general metamodel: profile-related elements, game description elements, scenario elements. Contrary to the EmoTED project, the scenarios elements have to be generated, not specified. Nevertheless we still have to specify the metamodel in order to detail the components of the three scenarios in terms of elements/properties and relations to be generated. From this metamodel, different models have to be considered:

- the game description model: it describes all the game elements (skills, resources or exercisers, in-game objects, etc.); it is an input model for the generator;
- the profile model: it describes the user's profile for one person; it is also an input model for the generator;
- the scenario model: it embeds the 3-dimensions global scenario (objective, structural and features scenarios); it is an output model of the generator.

Thus, we propose a model-driven, iterative and incremental process about the design of the adaptive generation of serious game scenarios. This process involves domain experts and computer scientists to conjointly design and validate the domain elements and rules relative to the generation of adapted scenarios.

4.2 An iterative and incremental process involving experts

Beyond the model-driven approach we also propose a dedicated process, Figure 5, to help specify and validate how the adaptive generation should work. Activities in italics, *meta-modeling* and *generator development*, are only performed by computer scientists because of the required expertise. Other activities involve both domain experts, i.e. non-technical users, and computer scientists.

- Game analysis and design: this activity aims at identifying and making explicit the various domain elements, properties, relations and also domain rules that are involved in the adaptive generation of the targeted scenarios; case-studies and other designs (diagrams, mock-ups, etc.) can also be explicit.
- Meta-modeling: this activity consists in specifying the metamodel that captures the static domain elements according to the model-driven framework presented in the previous subsection. We propose the use of the Eclipse Modeling Framework (EMF). Indeed, EMF is a modeling framework and code generator for building tools and other applications based on a structured data model (Steinberg et al., 2009).
- Generator development: this activity aims at developing the generator code that will take in charge the adaptive generation of the different scenarios. This development can be partially eased by using the EMF environment. Indeed, from our metamodel specification, EMF provides tools and runtime support to produce a set of Java classes for the models, along with a set of adapter classes that enable viewing and command-based editing of the model. Thus, the generator code can use the generated Java classes to import the input models, handle them to produce the scenario model, and then to serialize the output scenario.
- Modeling child's profile and game components model: domain experts and computer scientists specify both input models by using a basic tree-oriented editor, also generated by EMF from the metamodel specification.
- Generator execution: this activity consists in executing the generator code. It uses the child's profile model and game components model to generate a model embedding one, two or three of the inter-related scenarios. It can also used the other explicit designs (case-studies, mock-ups...). The generated scenarios can then be analyzed in order to verify their relevance, coherency and completeness in terms of elements required by the serious game, but also to validate the domain rules that drive this generation.

These activities are part of an iterative process. One can consider at least 3 iterations focusing respectively on the 3 incremental scenarios: objective scenario, then resources scenario, and finally the game scenario. Nevertheless other iterations can occur for a same targeted scenario according to the analysis of the results from the generator executions. Indeed, gaps between experts predictions and generated results can occur, if such predictions are feasible. More generally, the analysis of the generated scenarios can highlight some misunderstandings within the interdisciplinary team, or some misconceptions about the rules and elements involved in the generation. Some re-engineering iterations are then useful.

Note that changes in the meta-model will break

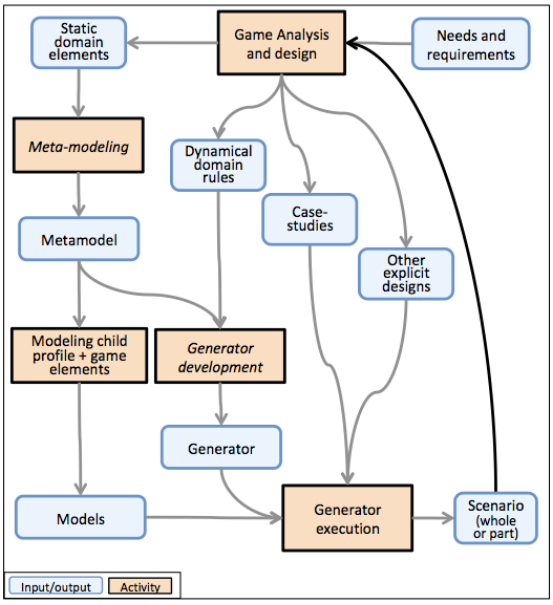

Figure 5: The iterative meta-modeling and modeling process involving domain experts.

compatibility of the models: it then requires to redo or modify them. Also, changes on the metamodel can impact the generator code: a new phase of Java classes generation can be required. If changes do not affect the metamodel but the semantic of the generation rules then only the generator code requires modifications.

5 Application to the case-study

In the next subsections the different steps of the process are described in more details by applying our proposition to the *Escape it!* context. We then present one of the use cases involving experts in order to validate our proposition (subsections 5.4).

5.1 The experts' knowledge

Several collaborative sessions with autism experts led us to progressively explicit:

- the global 'escape-the-room' gameplay adaptation,
- the visual performance skills in conformance with this gameplay,
- a more detailed description of the scene resolution in terms of objects, hiding elements, solution objects, etc.
- the domain rules to apply when generating a scenario.

Some of these generation rules, the elements from the profile and game components models in relation with them, are sketched in Table 1.

The generator we developed do not yet tackle the childs' profile elements for the *resource* and *game* levels of the generator (gray cells from Table 1). This choice allowed us to focus on the high-priorities elements and rules.

As mentioned in the table, some mapping rules have been made explicit in order to guide the generator in deciding how a scene is composed according to a specific difficulty level. For example, here are the mappings for the 'intermediary' level:

- some background elements can appear;
- some hiding objects can appear with 0 or several hidden objects inside according to their available slots;
- all selectable objects are in relation to the problem resolution (no other objects for disturbing purposes).

5.2 The Escape it! metamodel

The complete metamodel (an XML-based *ecore* file) is represented in a graphical form in Figure 6. It describes the concepts, properties and relations between concepts. It has been specified with the EMF framework.

This metamodel specifies all elements required for the generator as well as those that will be generated from it. It is not worth explaining all detailed elements. We then on purpose propose in Figure 7 a more abstract perspective of the different parts, and their relations, composing the metamodel.

There are 3 root elements in conformance with the 3 models to consider: *Game Description*, *Profile Elements* and *Scenario*. The *Game Description* is composed of 3 sub-parts: the *skills elements* (e.g. the visual skills), the *exercises elements* (e.g. the scenes and themes), the *game components* (the background, objects, locations, etc.) that are relative to a concrete exercise. Some elements from *Exercises* and *Game Components* parts will refer to some specific skills elements (e.g. the scenes must specified which targeted skills they can deal with). Similarly, the Scenario is composed of the 3 inter-related sub-parts: *Objective Scenario*, *Structural Scenario* and *Features Scenario* (each of them completing the previous one). The *Profile* part is not subdivided into other parts because, as mentioned in Table 1, we only tackled the profile elements required for generating the objective scenario.

Finally, the most interesting relations are the vertical ones: the generator will analyze the skills ele-

Table 1: The different domain rules and elements to deal with according to our 3x3-dimensions framework

	Game description	**User profile**	**Generation rules to consider for scenarios**
Objective scenario	- the visual skills to acquire - *dependency* relations between skills	- the acquired or in progress skills - their difficulty level - the number of levels to generate (n)	- only skills with *parents* at 'intermediary' level or higher are eligible
Structural scenario	- the themes (school, house, etc.) and the associated scenes (bedroom, classroom, etc.) - for each scene: the skills it can deal with	- the themes/scenes to exclude/favour according to each child's preferences/dislikes - information about previous game sessions	- generate n different scenes from the same theme if possible - else generate different scenes from various themes - else generate a minimum of the same scenes
Features scenario	- for each scene: the background elements, the hiding objects, the available object places, etc.	- for each scene: the objects to exclude/favour according to each child's preferences/dislikes - for each scene: information about elements involved in previous sessions	- mappings between each difficulty level and the objects to select and place into the scene

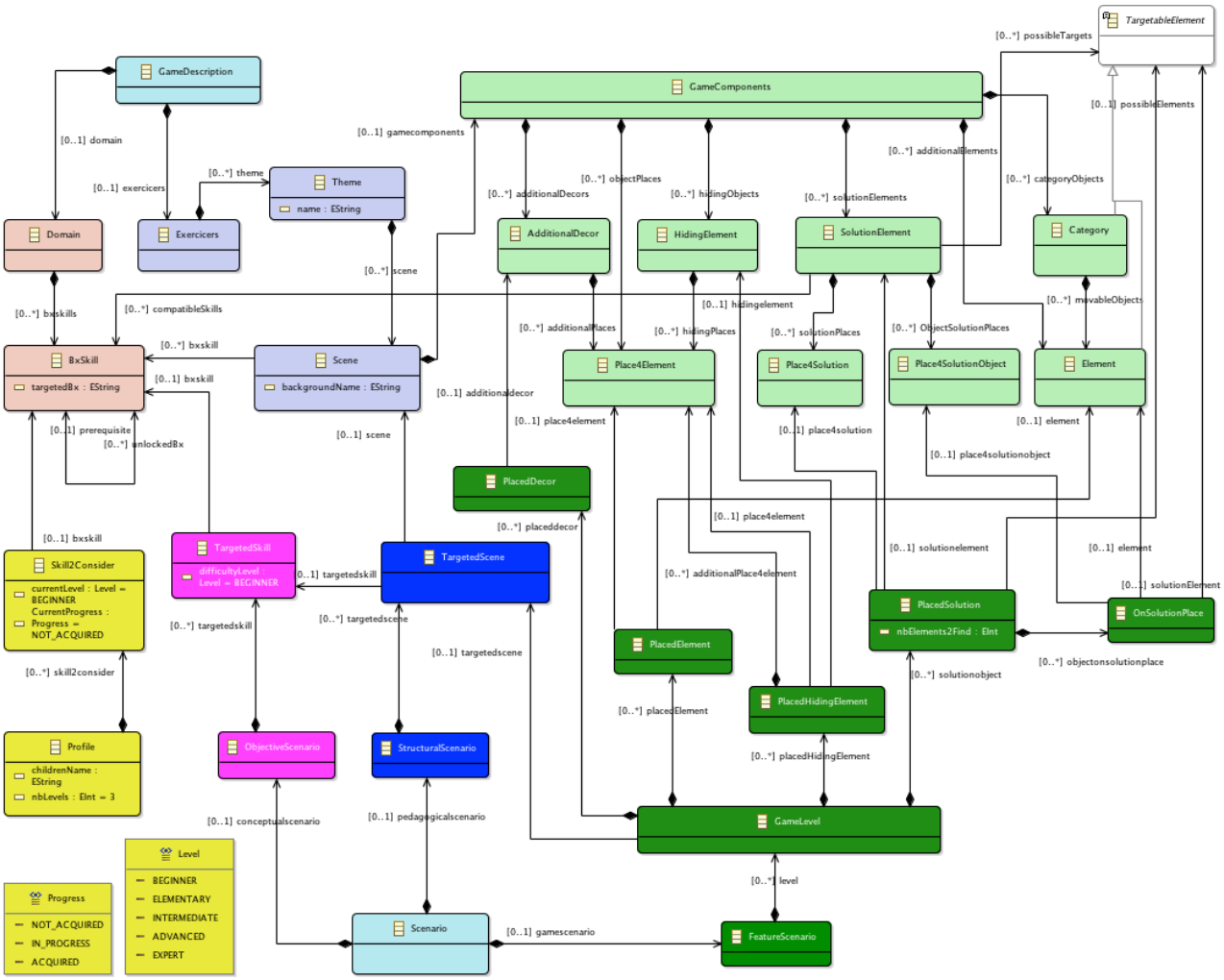

Figure 6: Complete view of the metamodel with variations of colors to discern the different dimensions.

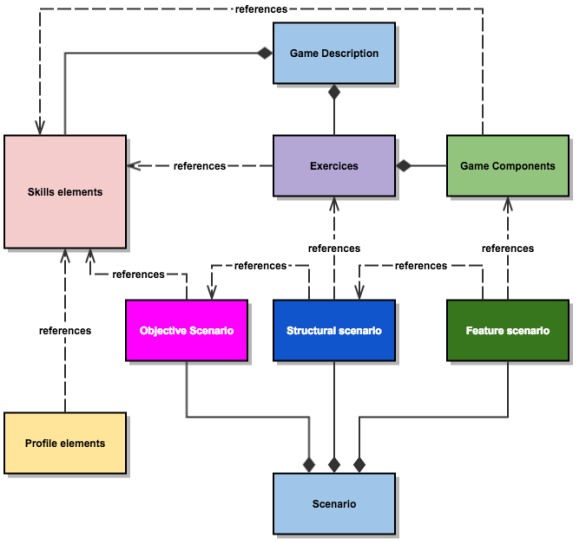

Figure 7: Conceptual parts and relations (*composed-of* and *references*) composing the metamodel. Similar colors are used between Figures 6 and 7 to ease the parts identification.

ments and profile elements in order to generate the objective scenario (including elements being related to skill elements). In a same way, the generator will then analyze the *Exercises elements* according to the previous identified skills to consider. Some of these exercises elements will be identified to be used by referencing them into the *Structural scenario* elements. Lastly, but the more complex part of the generator, the game components relative to the previous selected exercises elements are analyzed. It will allow the generator to create the appropriate *Features scenario* elements for specifying which objects have to appear in the scene and at which locations, according to the learning objective assigned to the scene.

5.3 Development of the generator

We developed the generator as a Java, console-user-interface program in order to benefit from the Java Model code and other code facilities generated from the metamodel. Nevertheless, the generation rules still have to be considered. Those corresponding to levels 1 and 3 (cf. Table 1) are hard-coded with aleatory procedures when several possibilities occur. We used the Java *Choco-Solver-4* library to specify level 2 (scenes and themes selection) rules as constraints-based programs.

As mentioned in previous sections we do not yet tackle the domain rules relative to the child's profile for levels 2 and 3. The generator allows the successive generation of the 3 scenarios. The resulting final scenario is persisted as a model that can be displayed and handled by the EMF-generated model editor. However we decided to display in the console a text-based readable print of the generated scenarios.

5.4 Validation of the scenario generator

This section is dedicated to the description of one use-case among those we experimented with domain experts in order to verify (whether the system is well-engineered and error-free) the generator and validate (whether the generator and generator rules meet the experts expectations and requirements) the generation rules. All those use cases were fictive but realistic according to experts suggestions. The modeled game components were limited to the one available in the prototype.

5.4.1 Modeling the input models

The generator requires two input models. The first one specifies the game components involved with the various scenario levels whereas the second one simulates a child's profile.

In our experiments with autism experts, the first model has been specified on the base of experts' requirements. We modeled the B3-B4-B8-B25 visual performance skills (respectively matching object to image, matching object to object, sorting categories of objects, making a seriation) and added their dependency relations (e.g. Figure 8 shows that the B3 skill unlocks the B4 and B8 skills, i.e. completing B3 at its highest difficulty allows to progress independently with the learning of the B4 and B8 skills). We then specified the description of the game *scenes* according to their relative *theme* (Figure 9). Finally, we specified the elements involved in the *gymnasium* scene (Figure 10): the one we focused on when developing the prototype.

This game description has been modeled using the tree-based editor generated from our metamodel thanks to the EMF tooling. Figures 8, 9 and 10 show different extracts of this unique model. The model root is a *Game Description* instance. The *composition* relations are naturally represented within this tree-based representation whereas properties and other relations are detailed in the *Properties view* according to the element currently selected.

In opposition to the *game components* model that is unique, several children profile models have been specified in order to test the correct use of generating rules about the difficulty progress. Figure 11 shows one of these profile models. It describes a child's profile wherein the B3 skill is acquired at its highest (*expert*) level. The B4 skill is at the *elementary* level, B8

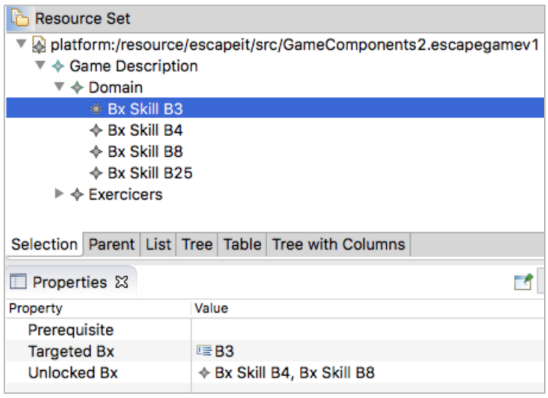

Figure 8: Partial view of the game description input model for level 1 (objective scenario).

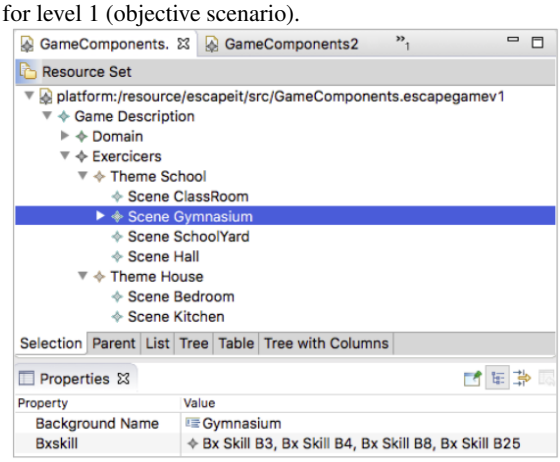

Figure 9: Partial view of the game description input model for level 2 (structural scenario).

is at the *intermediate* level, and B25 is at the *beginner* level.

5.4.2 Analyzing the output model

We only depict the output scenario generated from the child's profile example of the previous section.

The generator displays in the console user-friendly prints of the resulting scenario. First prints remind the input child's profile and the number of levels to generate. Then the objective scenario is displayed, followed by the additional information generated with the resource scenario (see Figure 12). For example, experts can verify that, for this very generation execution, 4 levels are proposed for the respective targeted and ordered skills: B4 / B25 / B8 and again B8 (with their difficulty level relative to the one specified in the child's profile). In this execution, the generator succeeded in proposing different scenes from the same theme (school).

The prints depicted in Figure 13 describe the details of the game scenario only for the third level involving the gymnasium scene (the only scene we fo-

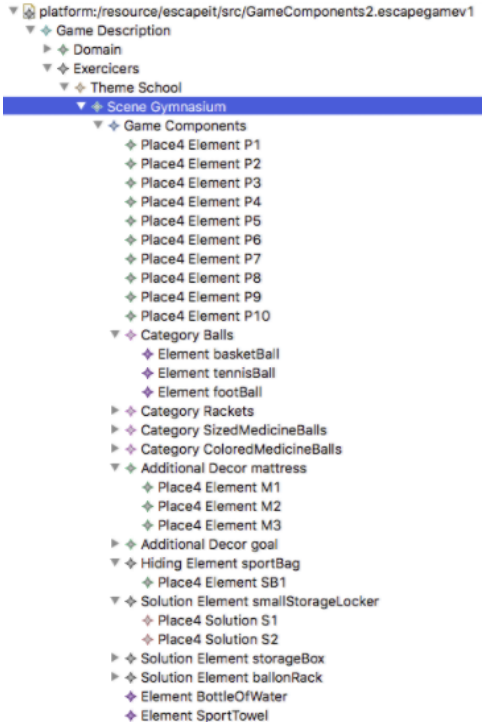

Figure 10: Partial view of the game description input model for level 3 (feature scenario).

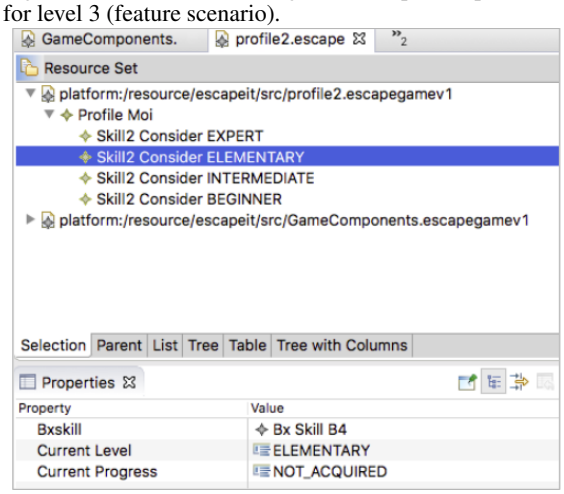

Figure 11: Partial view of a child's profile input model.

cused on). This description includes the objects used and the slots used within the scene. Some slots refer to the ones offered by background or hiding elements (cf. descriptions in Figure 10).

5.5 Validation of generating rules

Previous subsections described a case-study in order to present the different steps of our meta-modeling/modeling process. This case-study, with other ones, let us verify the correct functioning of the generator. It also allowed us to validate the rel-

```
******* INIT *********
Profile file loading...

******* Child Profile *******
This is the profile for: Moi
Skills:
- B3 (EXPERT)
- B4 (ELEMENTARY)
- B8 (EXPERT)
- B25 (BEGINNER)

nb levels to generate = 4

******* Conceptual Scenario *******
Targeted Skills:
- B4 (ELEMENTARY)
- B25 (BEGINNER)
- B8 (EXPERT)
- B8 (EXPERT)

******* Pedagogical Scenario *******
Targeted Scenes:
- SchoolYard - B4
- Hall - B25
- Gymnasium - B8
- ClassRoom - B8
```

Figure 12: Console prints after the generation of an adapted scenario: readable view of the *objective* and *structural* parts of the output model.

```
******* Game Scenario *******

>>>>>>>>>>>><<<<<<<<<<<<<<
SchoolYard level for the B4 skill, with difficulty ELEMENTARY
>>>>>>>>>>>><<<<<<<<<<<<<<
Hall level for the B25 skill, with difficulty BEGINNER
>>>>>>>>>>>><<<<<<<<<<<<<<
Gymnasium level for the B8 skill, with difficulty EXPERT
        - Additionnal decor element: mattress
        - Additionnal decor element: goal
        - Hiding element 'sportBag' on the place P1
        - Solution element 'storageBox' on the place S4 with
possible 3 target(s) of type 'Rackets'
        - Solution element 'storageBox' on the place S3 with
possible 6 target(s) of type 'Balls'
        - element 'pingpongRacket' on the place M1
        - element 'tennisRacket' on the place G1
        - element 'pingpongRacket' on the place G3
        - element 'tennisBall' on the place G2
        - element 'tennisBall' on the place P5
        - element 'footBall' on the place P3
        - element 'basketBall' on the place P9
        - element 'tennisBall' on the place SB1 / P1
        - element 'basketBall' on the place P8
        - element 'BottleOfWater' on the place M2
        - element 'SportTowel' on the place P7
        - element 'BottleOfWater' on the place P10
        - element 'BottleOfWater' on the place G4
        - element 'BottleOfWater' on the place P2
        - element 'SportTowel' on the place P4
>>>>>>>>>>>><<<<<<<<<<<<<<
ClassRoom level for the B8 skill, with difficulty EXPERT
```

Figure 13: Console prints after the generation of an adapted scenario: readable view of the *feature* part of the output model.

evance of the generation rules with domain experts. We now give some additional information about this validation.

We conducted two collective sessions for exploiting our generator with two experts. The first session focused on the the first two layers: *objective* scenario and *resource* scenario. We already tested our generator with various case-studies before this meeting in order to avoid coding issues. This session consisted in modeling some case studies with experts and analyzing the generated scenarios. During this session we decided not to take into account an initial domain rule (not mentioned in Table 1) about generating 80% of levels with skills at a maximal difficulty level 'intermediary' against 20% of skills with higher difficulty levels. They finally did not consider it relevant. In contrast we listed different mappings between every difficulty level and the game objects to consider at the *game* scenario level.

The second collaborative session focused on the generation of the game scenario. Before, we modified the game components model to specify the gymnasium scene of the school theme as the only available scene. Thus, it forced the generator to select this scene and then detail its game objects when dealing with the game scenario generation. This session let us validate the good matching between the difficulty levels and the different game rules. The fictive child's profiles were only used here to drive the focus on a targeted skill, one-by-one, by experimenting all difficulty levels mappings. In order to better visualize the results proposed by the generator we used some paper-based prints of the scene's background (with annotations about slots), objects, solution objects, hiding objects, etc. We read the console-based generated result and placed the mentioned objects in their slots. When done, we discussed the relevance of the proposition with experts.

These very first validation sessions confirmed our proposition to help domain experts in designing and validating the required rules to drive the generation of adapted learning scenarios.

6 CONCLUSIONS

In this paper we studied the problem of adapting learning game scenarios to people with specific needs. We presented a Model-Driven Engineering (MDE) framework that ease the design and validation of adaptive learning scenarios. It tackles the personalization issue by helping domain experts and computer scientists in making explicit and validating the domain elements and rules involved in the adaptation. The proposition is based on a metamodel specifying at first the domain elements according to both a 3-incremental-perspective on the resulting scenario, and a 3-dimensions specification of domain elements to use and produce. The framework proposes to model the game description and the child's profile as input models for the generator that will produce the adapted scenario as an output model. We applied the framework in the context of the *Escape it!* project that aims at helping young children with Autistic Syndrome Disorder to learn and generalize visual performance skills. Collaborative sessions with autism ex-

perts have been conducted.

First results emphasized the interest in validating the adaptation elements and rules on the early design phases. It could be useful to avoid the re-engineering cost when adaptation rules are invalidated when testing a prototype. Moreover, MDE tools provide some facilities in order to drive and ease the development of a basic console-based generator. Models are both executable by the generator, and human-readable when specified with a tree-based model editor (generated by EMF from the metamodel specification). It then offers a support for conducting collaborative design sessions between computer scientists and domain experts.

Our very first validation sessions confirmed our proposition to help domain experts in designing and validating the required rules to drive the generation of adapted learning scenarios. Further developments will focus on integrating the generator output models (XML file) in our Unity-based game prototype. It will allow us to propose more accurate validations with respect to the prototype independence to changes on the generator. We also highlighted that the dynamical domain rules, like the generation rules and the mapping rules between the difficulty levels and the game objects involved within a scene resolution, are not explicit: they are hard-coded in the generator. Because it can also lead to coding issues, further research works about specifying such informations have to be undertaken. We have started tackling this issue by experimenting various MDE techniques: models transformations, model weaving, validation of models in conformance to rules written with the Object Constraint Language (OCL), etc.

REFERENCES

Callies, S., Sola, N., Beaudry, E., and Basque, J. (2015). An empirical evaluation of a serious simulation game architecture for automatic adaptation. In *R. Munkvold & L. Kolas (eds.), Proceedings of the 9th European Conference on Games Based Learning (ECGBL 2015)*, pages 107–116. Reading, UK: Academic Conferences and Publishing International Limited.

Deterding, S., Dixon, D., Khaled, R., and Nacke, L. (2011). From game design elements to gamefulness: Defining "gamification". In *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, MindTrek '11, pages 9–15, New York, NY, USA. ACM.

Ern, A. (2014). The use of gamification and serious games within interventions for children with autism spectrum disorder.

Hussaan, A. M. and Sehaba, K. (2016). *Consistency Verification of Learner Profiles in Adaptive Serious Games*, pages 384–389. Springer International Publishing, Cham.

Janssens, O., Samyn, K., Van de Walle, R., and Van Hoecke, S. (2014). Educational virtual game scenario generation for serious games educational virtual game scenario generation for serious games. In *Proceedings of the IEEE 3rd International Conference on Serious Games and Applications for Health (SeGAH'14)*.

Laforcade, P. and Vakhrina, V. (2015). *A Domain-Specific Modeling approach for a simulation-driven validation of gamified learning environments - Case study about teaching the mimicry of emotions to children with autism*.

Leaf, R. B. and McEachin, J. (1999). *A Work in Progress: Behavior Management Strategies and a Curriculum for Intensive Behavioral Treatment of Autism*. New York: DRL Books.

Loiseau, E., Laforcade, P., and Mawas, N. E. (2015). Turning recurrent uses of e-learning tools into reusable pedagogical activities - a meta-modeling approach applied to a moodle case-study. In *Proceedings of the 7th International Conference on Computer Supported Education*, pages 64–76.

Lopes, R. and Bidarra, R. (2011). Adaptivity challenges in games and simulations: A survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(2):85–99.

Mussbacher, G., Amyot, D., Breu, R., Bruel, J.-M., Cheng, B. H., Collet, P., Combemale, B., France, R., Heldal, R., Hill, J., Kienzle, J., Schöttle, M., Steimann, F., Stikkolorum, D., and Whittle, J. (2014). The Relevance of Model-Driven Engineering Thirty Years from Now. In Dingel, J., Schulte, W., Ramos, I., Abraho, S., and Insfran, E., editors, *Model-Driven Engineering Languages and Systems*, volume 8767 of *Model-Driven Engineering Languages and Systems*, page 18, Valencia, Spain. Springer International Publishing Switzerland.

Partington, J. and Analysts, P. B. (2010). *The Assessment of Basic Language and Learning Skills-revised (the ABLLS-R)*.

Sehaba, K. and Hussaan, A. (2013). Goals: generator of adaptive learning scenarios. In *International Journal of Learning Technology*, volume 8, pages 224–245.

Sina, S., Rosenfeld, A., and Kraus, S. (2014). Generating content for scenario-based serious-games using crowdsourcing. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 522–529. AAAI Press.

Steinberg, D., Budinsky, F., Paternostro, M., and Merks, E. (2009). *EMF: Eclipse Modeling Framework 2.0*. Addison-Wesley Professional, 2nd edition.

Whyte, E. M., Smyth, J. M., and Scherf, K. S. (2015). Designing serious game interventions for individuals with autism. *Journal of Autism and Developmental Disorders*, 45(12):3820–3831.

Zakari, H. M., Ma, M., and Simmons, D. (2014). *A Review of Serious Games for Children with Autism Spectrum Disorders (ASD)*, pages 93–106. Springer International Publishing, Cham.