Generation of adapted learning game scenarios: a Model-Driven Engineering approach

Pierre Laforcade[1] and Youness Laghouaouta[1]

Computer Laboratory of Le Mans University, France
{pierre.laforcade, youness.lahouaouta}@univ-lemans.fr

Abstract. Adaptativity is a key-concern when developing serious games for learning purposes. It makes it possible to customize the game according to each learner individuality. To deal with adaptativity, this chapter proposes a Model-Driven Engineering approach that supports dynamic scenarization instead of implementing fixed configurations of learning scenarios. The base principle is to consider the generation of scenarios as a model transformation of a learner profile and a game description models toward adapted scenarios. This proposal has been applied to the context of the *Escape-it!* research project that aims to propose an "escape-room" game for helping children with Autistic Syndrome Disorder (ASD) to learn visual performance skills.

Keywords: Serious game, autism, learning scenarios, adaptation, model driven engineering.

1 Introduction

The use of serious games [3] in Autistic Syndrome Disorder (ASD) interventions has become increasingly popular during the last decade [4]. 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 [19].

This chapter 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 uses 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 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. Our proposal to deal with this issue is based on Model-Driven-Engineering (MDE) principles and tools. MDE [14] is a research domain promoting an active use of 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 [11] however the learning scenarios were specified by teachers and not generated by machine.

The remainder of this chapter is organized as follows. The next section presents the *Escape it!* project contextualizing our research. Section 3 discuses related work. Our model-driven based approach for the adaptive generation of learning scenarios is presented in Section 4. Section 5 illustrates the application of our proposal and discusses the obtained results. Finally, Section 6 concludes this chapter and presents the next directions for further research.

2 Context and motivation

This section presents the *Escape it!* project that contextualizes our research. Before presenting the main gameplay we draw the project rationale.

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 [15]) : matching an object to another identical object, sorting objects into different categories, making seriation of objects, etc. 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 and 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 [4][20].

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, guidance, feedback, reinforcements 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 information for our concern:

- targeted skills: a subset from the 27 visual performance skills depicted in the ABLLS-R, the relevant ones for a mobile game-play 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.

Figure 1 depicts an example of a scene which targets the B8 skill (i.e. sort non-identical items) in the '*Expert*' difficulty level. Various trucks and balls have to be found and moved into the appropriate storage boxes before the door opens. Interactive hiding places, like the closet and its drawer, can be opened showing hidden objects.

Figure 2 illustrates an example of another scene which targets the B13 skill (i.e. matching a sequence of objects) in the '*Intermediate*' difficulty level. Mustard and ketchup dispensers, and coke bottles have to be found and moved into their appropriate locations in the middle fridge door shelf with the aim of matching the sequence on the top shelf. Some of the objects to found are already visible, others can be hidden in the shopping bags. Note that experts validated objects and gameplay in order to avoid inappropriate interactions or explicit actions that children do not normally realize in everyday life.

Figure 3 shows a bedroom scene variant which targets the B3 skill (i.e. matching identical objects) in the '*Elementary*' difficulty level. Dinosaurs toys have to be found and moved in the top shelf of the drawer wherein an exemplar is already visible. The desk is an example of randomized additional decor that can provide new locations for placing objects. In Figure 3 two dinosaurs toys are placed in potential locations provided by the desk.

Fig. 1. An example of the *bedroom* scene

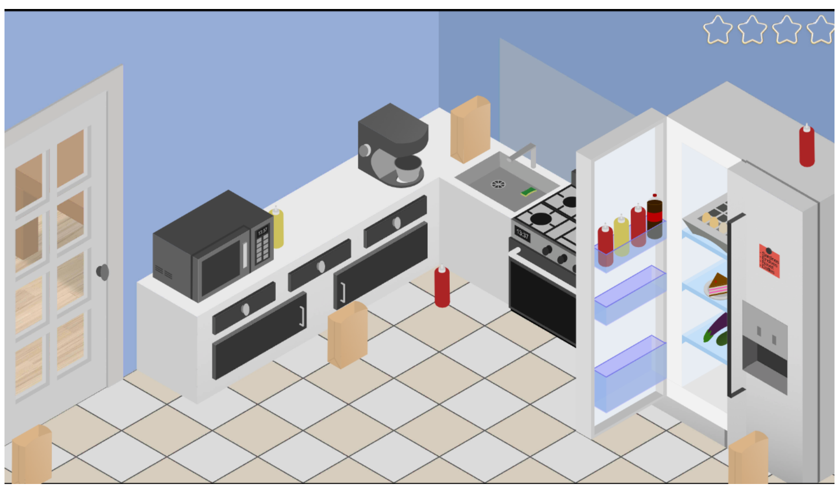

Fig. 2. An example of the *kitchen* scene

2.3 Anatomy of a scene

Whichever scenes are selected for the learning scenario, they share common features:

- a background image that depicts a familiar scene for children, with a few recognizable objects;
- several empty slots where objects to find can be later positioned.

Fig. 3. An example of the *bedroom-2* scene

– 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 place a solution object 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.

2.4 Motivation

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. . .) 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 [1]. 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 to dynamically generate game sessions adapted to each child's profile.

3 Related work

The adaptation of Technology-Enhanced Learning (TEL) systems allows the personalization of learning by adapting whole or parts of the presented systems (contents, resources, activities. . .) to the learner's needs, interests and abilities. An adaptable learning system is generally considered as a learning system that can be manually configured by its end-users. By contrast, adaptive learning environments aim at supporting learners in acquiring knowledge and skills in a particular learning domain while being automatically adapted to the changing needs of the learner. In our context, we are not interested in the adaptability of the mobile learning game for children with ASD but on the adaptivity of the game (i.e. to provide adapted game sessions in accordance to the children profiles).

3.1 Generation of adaptive scenarios

The motivation for steering adaptivity in serious games is to improve the effectiveness of the knowledge transfer between the game and its players. Several studies tackled the adaptation issue in order to find a balance between the player's skills and the game challenge level. The learning goals to achieve are usually strongly coupled with the gradual personal improvement of a skill set. Generally, adaptive serious games have specialized *ad hoc* approaches where game components are adjusted in order to encourage training of a specific skill.

Research work dealing with adaptivity have different targets: game worlds and its objects, gameplay mechanics, nonplaying characters and AI, game narratives, game scenarios/quests. . . They also rely on various methods: bayesian networks, ontologies, neuronal networks, rules-based systems, procedural algorithms. . . [17][7][2]. Game scenarios are generally defined as *the global progression within a game level, its initial settings and the logical flow of events and actions that follow* [5], whereas game worlds are *the virtual environments within which gameplay occurs.* In our context, our interest is about learning game scenarios, because each scene to achieve targets a specific skill, while disregarding the flow of events or actions. The resolution of a scene only requires that the learners find and move objects to their appropriate target locations. Our context partially maps the game world and its object definition in the way that the available objects of scenes can have zero or more instances according to the generation process.

Reaching beyond skill-driven adaptivity and integrating scenario with world adaptation/generation while the game is running remains a research challenge [12]. There are two approaches to tackle it: 1) during the loading stage of a game session by considering player-dependent informations, and 2) in real-time during game playing. Our concern relies on the first approach: scenarios are generated before starting every new game sessions (the 3-to-5 levels).

In [13], the authors have proposed a system for generating content highlights the involvement of domain experts (i.e. teachers) to control the content generation. Teachers can select pre-created game objects, add new learning content to them and create relationships between objects. Knowledge about objects and their relationships seems a basis for solving and generating all the appropriate content. It could be a valuable contribution to control the generation of our learning game scenarios by using knowledge on the objects of each scene and their relationships. Such game knowledge should be specified at a high semantic level in order to involve domain experts. That approach is very interesting in our context. We could specify every game scenes (bedroom, kitchen. . .) in a descriptive way of all available slots, additional decors, selectable objects. . . (cf. elements listed in Section 2.3), as well as their restrictive relations (i.e. solution objects).

3.2 An architecture and approach for generating scenarios

Closer to our concerns, the work presented in [16] proposes a generic architecture for personalizing a serious game scenario according to learners' competencies and interaction traces [6]. The architecture has been evaluated with the objective to develop a serious game for evaluating and rehabilitating cognitive disorders. It is organized in three layers: domain concepts (i.e. the domain-specific concepts and their relations), pedagogical resources (i.e. "*any entity used in the process of teaching, forming or understanding, enabling learning or conveying the pedagogical concepts*" [16]) and game resources (i.e. "*either static objects or those endowed with an interactive or proactive behavior according to the game*"). Machine learning is used to update the learner profile based on interaction traces.

In addition, this proposal allows the generation of three successive scenarios (conceptual, pedagogical and serious game scenarios) according to the three presented layers. As for the validation of the generated scenarios, the authors used an evaluation protocol. For that, experts were involved at first to validate the domain rules, *a priori* of the generator implementation, and then to produce scenarios for specific contexts. These scenarios are compared to the generated ones. Hence, experts guide the requirements specification and validation activities, but they are not directly involved in the generation process.

4 A model-driven approach to support adaptive generation of scenarios

4.1 Overview of the approach

Our proposal is based on the idea to combine the general architecture of a scenario generator from the CLES project [16] with the Model-Driven Engineering approach and process used in the EmoTED project [10] 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:

- 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, this 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. In the *Escape it!* context, this concerns 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, this 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 information to adapt the game sessions to a user's profile. Information about how these descriptions elements are functioning (static and dynamical dimensions) might be addressed by the generator and the serious game engine: they are out the scope of the scenarization process.

From the EmoTED project [10], we follow the metamodeling/modeling approach using the EMF framework (*Eclipe Modeling Framework*)[18]. 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 (see Figure 4). 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. . .). It is an input model for the generator.
- the profile model: it describes the user's profile for one child. 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.

The structuring of the elements related to the game as well as generation rules of learning scenario rely on the ASD experts recommendations/requirements. The following section details this aspect.

4.2 Game analysis

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

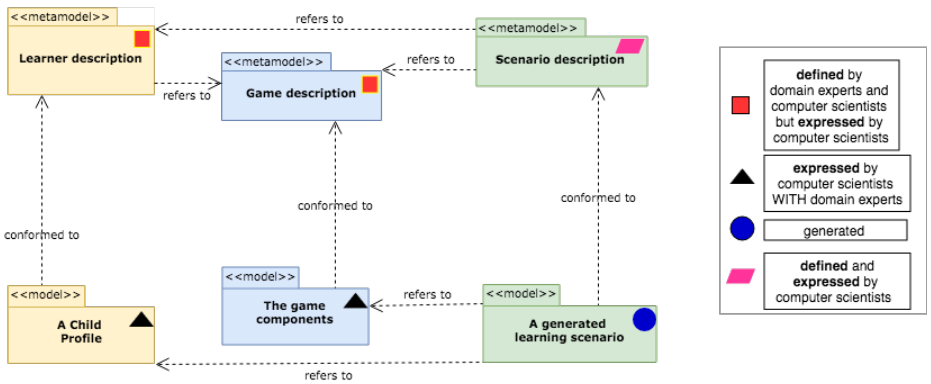

Fig. 4. The proposed 3x3 metamodel-based architecture.

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

Some of these generation rules as well as the elements from the profile and game components models in relation to them, are sketched in Table 1. The generator we developed do not yet tackle the child's 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 Table 1, 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 objects for disturbing purposes).

Some mapping rules have also been established to guide scenes construction according to a specific difficulty level (cf. Table 2). Additional information like the different ranges according to the difficulty level have been specified.

4.3 Metamodeling architecture

The domain elements and relations required for the adaptive generation of scenarios are structured according to three metamodels (i.e. the *Profile*, *Game Description*, and *Scenario* metamodels). We have used the EMF platform[1] to express the relevant metamodels (see Figure 5). We have to notice that Figure 5

[1] http://www.eclipse.org/modeling/emf/

Table 1. The different domain rules and relevant elements according to our metamodeling architecture (from [9]).

	Game description	**User profile**	**Generation rules for scenarios**
Objective scenario	-visual skills to acquire. -*dependency* relations between skills.	-acquired or in progress skills. -their difficulty level. -number of levels to generate.	-only skills with *parents* at '*Intermediate*' level or higher are eligible. -80% of targeted skills with a difficulty level less than '*Intermediate*'.
Structural scenario	-themes and associated scenes. -skills targeted by each scene.	-themes/scenes to exclude/favour according to child's preferences/dislikes. -history of proposed scenes.	-generate different scenes from the same theme.
Feature scenario	-background elements, hiding objects, available object places of each scene.	-scene objects to exclude/favour according to child's preferences/dislikes. - objects involved in previous sessions.	-mappings between each difficulty level and the objects to select and place into the scene.

Table 2. Difficulty levels and their impacts on the generation process.

	Additional decors	Hidding objects	Number-of-objects-to-place indicator on solution objects	All movable objects are parts of the solution	Range of objects to find
Beginner	no	no	yes	yes	low
Elementary	yes	no	yes	yes	low
Intermediate	yes	yes	yes	yes	medium
Advanced	yes	yes	no	yes	medium
Expert	yes	yes	no	no	large

depicts all related constructs as one metamodel for better comprehending the inter-metamodels references.

We have to notice that each scenario's perspective (i.e. objective, structural and features) has been considered when defining the implied metamodels (see Figure 6). For example, the generation of an objective scenario considers a relevant subsets of the profile elements (e.g. skills and their levels for a specific child) and the game description elements (the ones representing the skills that are tackled by the game).

A *Scenario* instance contains three inter-related elements: objective, structural and feature scenarios. By following the same decomposition approach, the *Game Description* constructs are decomposed into three subsets that match the scenario's perspectives: the skills elements (visual skills), the exercises elements (scenes and themes) and the game components associated with a concrete exercise (background, objects, locations. . .). Some elements from *Exercises* and *Game Components* parts will refer to specific skills elements (e.g. scenes must specify which targeted skills they can deal with). As for the *Profile* constructs, they are limited to elements required for generating the objective scenario. The remaining perspectives are not yet handled by our proposal (they are highlighted with grey color in Table 1).

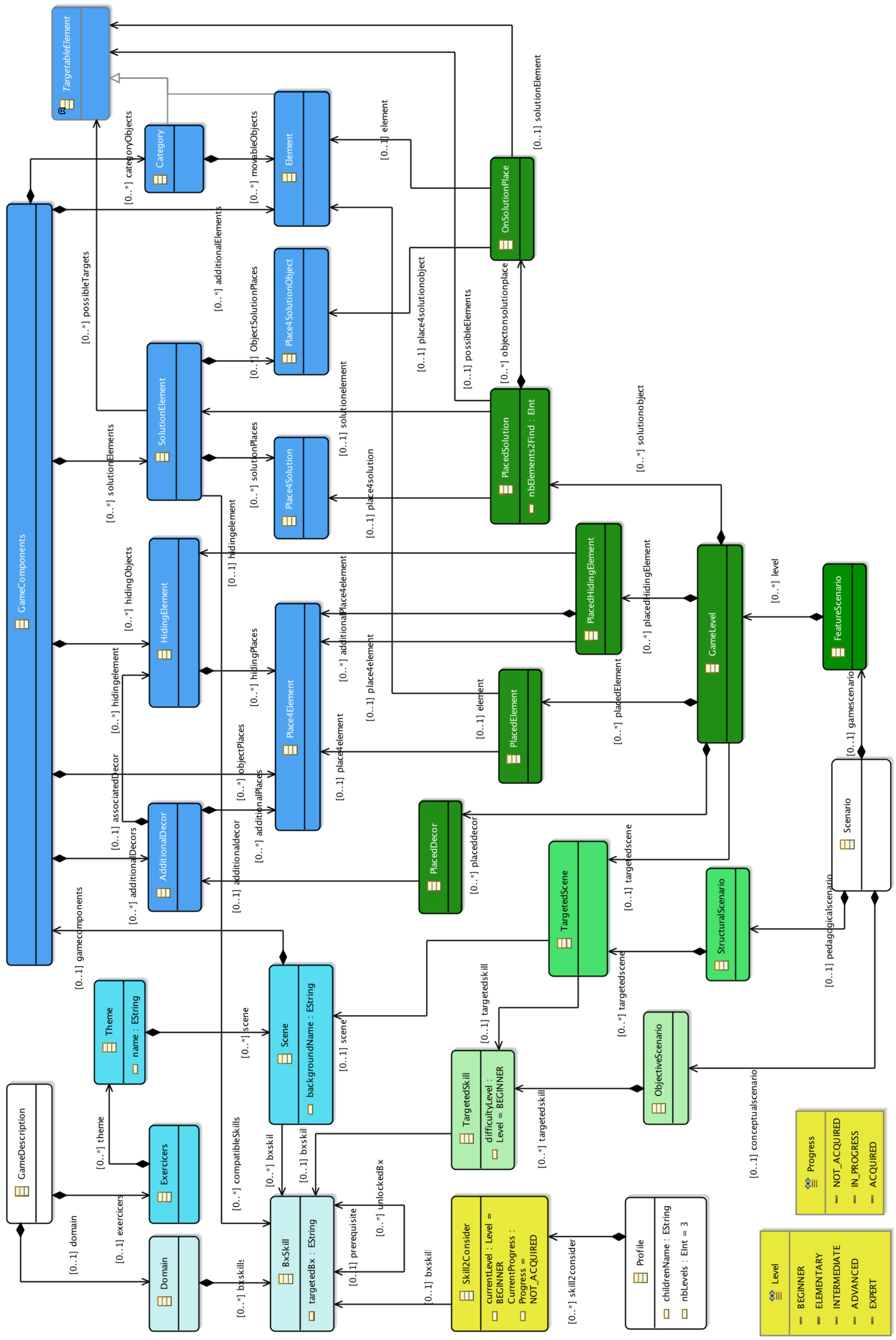

Fig. 5. Complete view of the metamodels with variations of colors to discern the different dimensions/perspectives.

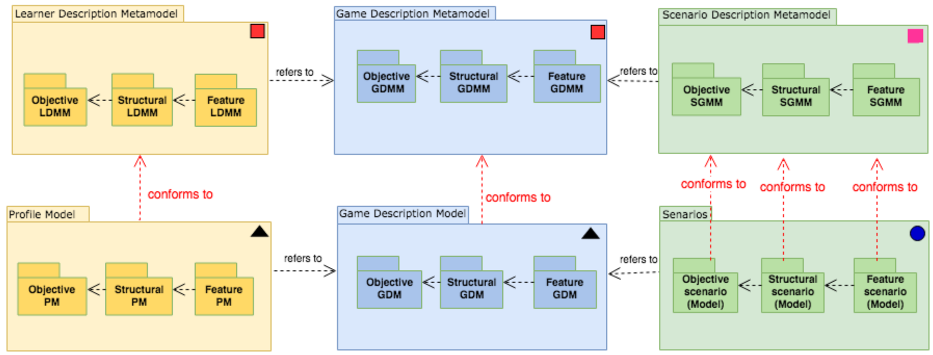

Fig. 6. The detailed proposed 3x3 metamodel-based architecture.

4.4 Generation of learning scenarios

The generation of scenarios adapted to child profiles is implemented as a model transformation written in Java/EMF. It uses the profile and game description models as inputs to allow the successive generation of the three perspectives of an adapted scenario (cf. Figure 7). The generation rules are then hard-coded with randomness when several choices are met. It is worth noting that the experts requirements related to dynamic domain rules are not easy to implement. In fact, the implemented model transformation uses sometimes an external constraints solving library to tackle some very specific generation steps.

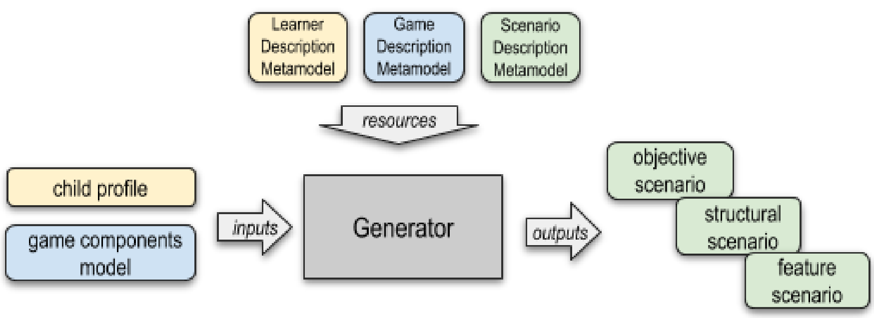

Fig. 7. Conceptual view of the transformation.

Employing the transformation presented above performs the generation of adapted scenarios. However, interpreting the generated models using basic EMF editors is not appropriate to perform domain rules validation. As a solution, we have implemented a support for integrating the generated scenarios in the learning game prototype (developed using Unity[2]). This concerns the low level

[2] https://unity3d.com/

scenario (i.e feature scenario) and makes it possible to visual and play the corresponding scenes in order to carry out effective tests of the game. It allow us to propose more accurate validations with respect to the prototype independence to changes on the generator. The scene depicted in Figures 1, 2 and 3 were generated using the proposed integration support.

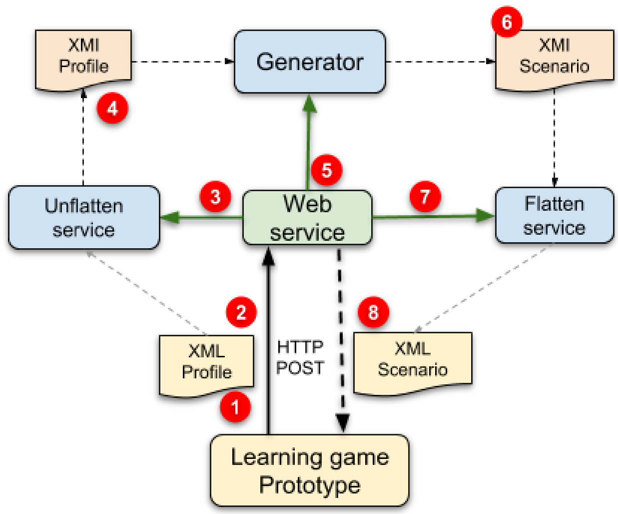

Fig. 8. Integration of the generator service into the learning game.

The main technical key points of this integration are illustrated in Figure 8:

1. The learning game serializes the profile of the current log in user as an XML-based file.
2. The learning game makes a Web service call using an HTTP POST request in order to upload the profile file.
3. The Web service executes at first an *unflatten* service for getting the profile file as a model conformed to our profile metamodel. Indeed, the generator requires the profile input model in a very specific format. Skills do not have to be declared in the profile: the profile have to reference skills defined in the *Game Description* input model. This mapping is concretely realized by the mean of an MDE model transformation using the *Epsilon Transformation Language* (ETL) [8].
4. The target profile is now in an XMI format in conformance to the profile metamodel.
5. The Web service calls the generator that makes use of the new conformed profile file (with all other model and metamodels as mentioned in Figure 7 that are already located on the Web service side).
6. The generation produces the output scenario model.

7. The Web service executes a *flatten* service. It has the opposite objective of the unflatten service: current scenario makes a lot of references of game elements (objects, decors. . .) from the *Game Description* model. The scenario must be self-sufficient for being handled by the game prototype. A dedicated ETL model transformation performs this task.
8. The resulting XML scenario is returned by the Web service to the prototype. This scenario will be parsed and used in the learning game.

5 Validation

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

5.1 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 based on the experts' requirements. We modeled the B3-B4-B8-B9-B13-B19-B25 visual performance skills (respectively matching object to object, matching object to image, sorting non-identical items into categories, placing objects on their marks, sequencing pattern to match visual model, sorting by feature, making a seriation - ordering by size, shapes...) and added their dependency relations (e.g. Figure 9 shows that the B3 skill unlocks the B4 and B8 skills, i.e. completing B3 at least at a sufficient 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 10). Finally, we specified the elements involved in the different scenes. As an example, Figure 11 depicts the detailed description of the *gymnasium* scene.

This game description has been modeled using the tree-based editor generated from our metamodel. Figures 9, 10 and 11 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 description* model that is unique, several children profile models have been specified in order to test the correct application of the generation rules related to the difficulty progress. Figure 12 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 is at the *intermediate* level, and all other skills are at the *beginner* level.

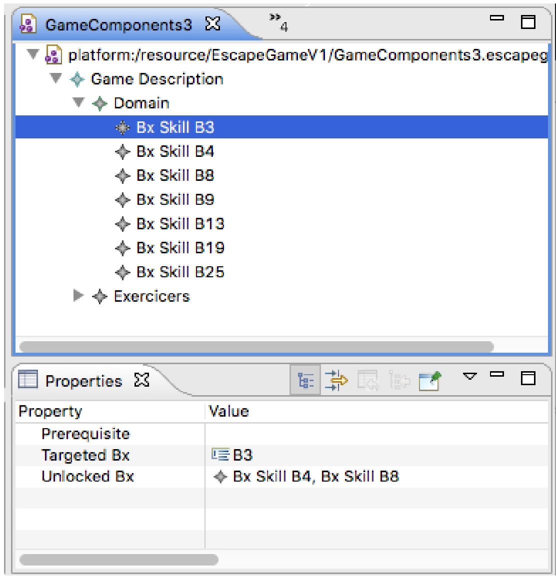

Fig. 9. Partial view of the game description input model for level 1 (objective perspective).

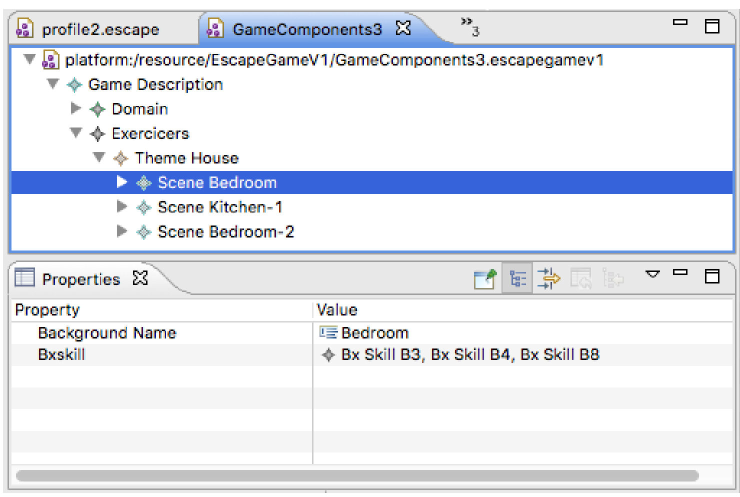

Fig. 10. Partial view of the game description input model for level 2 (structural perspective).

5.2 Analysis of the generated scenario

We only depict the output scenario generated from the child's profile described in 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

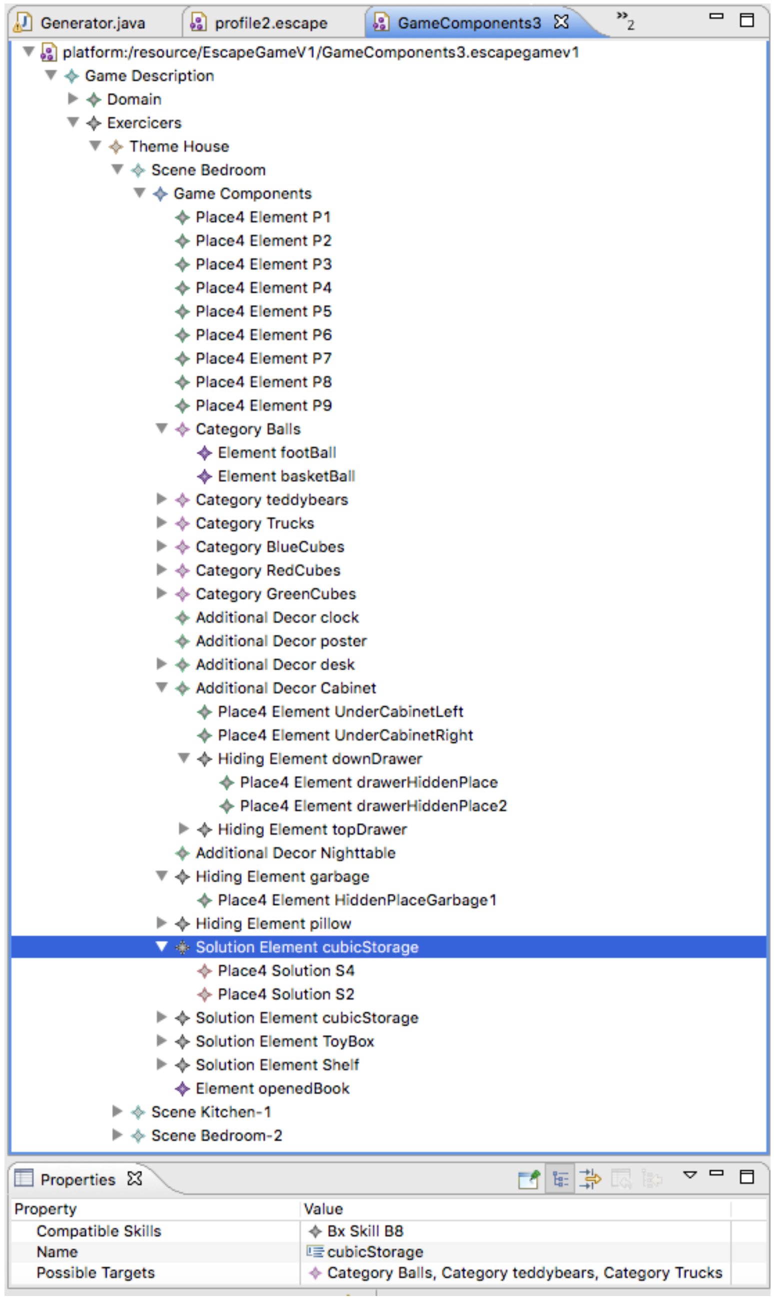

Fig. 11. Partial view of the game description input model for level 3 (feature perspective).

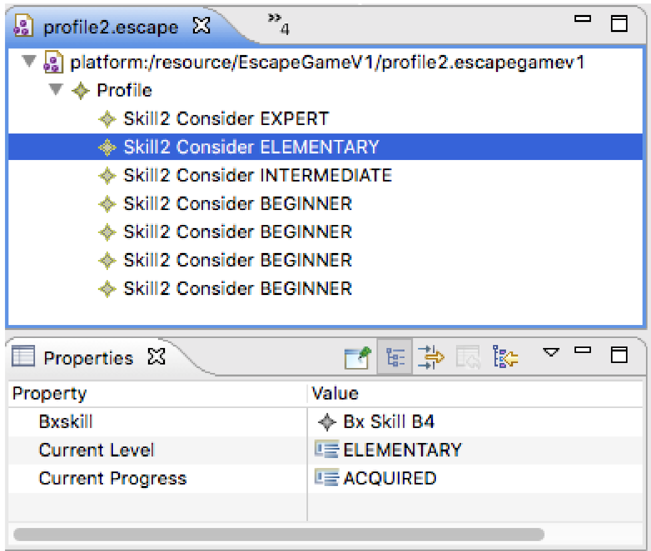

Fig. 12. Partial view of a child's profile input model.

to generate. Then the objective scenario is displayed, followed by the additional information generated with the resource scenario (see Figure 13). For example, experts can verify that, for this very generation execution, 4 levels are proposed for the respective targeted and ordered skills: B25 / B4 / B8 and B25 again (with their difficulty level corresponding to the one specified in the child's profile). In this execution, the generator succeeded in proposing different scenes from the same theme (*home*).

As for the examination of the third level scenario (i.e. feature scenario), we used the integration support based on Unity (c.f. Section 4.4) which provides a playable prototype of the corresponding related game level. The integration architecture presented in Figure 8 is used: Figure 14 shows the resulting XML scenario of the 8-point. This scenario is then parsed and used by the Unity game in order to set-up the successive 4 levels. Finally, Figure 15 illustrates the matching in-game scene for the second B4 level.

5.3 Validation of generating rules

We have conducted several collective validation sessions with two ASD experts. The proposed MDE based approach allowed us to varying situations proposed to domain experts without significant effort. Indeed, we have expressed several profiles and apply the same transformation to automatically generate the consequent scenarios.

As a first feedback, the experts decided to disregard the 80/20 generation rule. This rule stipulates that 80% of the skills referenced by the generated *objective* scenario must be at a difficulty level less than '*Intermediate*' against

```
******* INIT *********
Profile and game description models loading...
DONE

******* Child Profile *******
This is the profile for: Tom
Skills:
- B3 (EXPERT)
- B4 (ELEMENTARY)
- B9 (INTERMEDIATE)
- B13 (BEGINNER)
- B8 (BEGINNER)
- B19 (BEGINNER)
- B25 (BEGINNER)

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

******* Structural Scenario *******
Targeted Scenes:
- Bedroom-2 - B25
- Bedroom - B4
- Kitchen-1 - B8
- Bedroom-2 - B25
```

Fig. 13. Console prints after the generation of an adapted scenario: readable view of the *objective* and *structural* parts of the output model.

20% at higher level. Indeed, the experts realized that this rule cannot be satisfied in all possible cases (basically for children not familiar with the game and those at an advanced stage). On the other hand, the experts have proposed new rules concerning the selection of candidate scenes about the *structural* scenarios. The base principle is to diversify the scenes offered to the child while trying to use the same theme.

Another collaborative session focused on validating the good matching between the difficulty levels and the different game rules. Some of these rules are depicted in Table 2. Other rules detail for each skill and each difficulty level, how many solution objects and movable objects can be instantiated. For example the *B8* - sorting - skill at the *beginning* level requires one solution object and one object to find, whereas the same skill at *expert* level requires 2 solution objects with a different random number of corresponding movable objects for each solution object (from 0 to the number of available spots specified for the scene).

```xml
<?xml version="1.0" encoding="ASCII"?>
<flatModel4Unity:Game xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:
flatModel4Unity="flatModel4Unity">
  <level scene="Bedroom-2" nbElementsToPlace="1" difficulty="BEGINNER">
    <element pos="P2" name="CubeBlue2"/>
    <solutionobject pos="Pos4ShelfB25" name="ShelfB25" acceptedElements="
    CubeBlueB25" targetedSkill="B25" nbSol2Find="1">
      <element pos="Shelf1" name="CubeBlue5"/>
      <element pos="Shelf2" name="CubeBlue4"/>
      <element pos="Shelf5" name="CubeBlue1"/>
      <element pos="Shelf3" name="CubeBlue3"/>
      <solutionarea pos="Shelf4" acceptedElement="CubeBlue2"/>
    </solutionobject>
  </level>
  <level scene="Bedroom" nbElementsToPlace="2" difficulty="ELEMENTARY">
    <placeddecor name="desk">
      <element pos="D2" name="basketBall"/>
    </placeddecor>
    <element pos="P8" name="basketBall"/>
    <solutionobject pos="S2" name="cubicStorage" acceptedElements="basketBall"
    targetedSkill="B4" nbSol2Find="2"/>
  </level>
  <level scene="Kitchen-1" nbElementsToPlace="1" difficulty="BEGINNER">
    <element pos="B5" name="RedPlate"/>
    <solutionobject pos="SolTopPlates" name="WallCupboardOpenedPlatesSol"
    acceptedElements="Plates" targetedSkill="B8" nbSol2Find="1"/>
  </level>
  <level scene="Bedroom-2" nbElementsToPlace="1" difficulty="BEGINNER">
    <element pos="P6" name="CubeBlue3"/>
    <solutionobject pos="Pos4ShelfB25" name="ShelfB25" acceptedElements="
    CubeBlueB25" targetedSkill="B25" nbSol2Find="1">
      <element pos="Shelf5" name="CubeBlue1"/>
      <element pos="Shelf1" name="CubeBlue5"/>
      <element pos="Shelf4" name="CubeBlue2"/>
      <element pos="Shelf2" name="CubeBlue4"/>
      <solutionarea pos="Shelf3" acceptedElement="CubeBlue3"/>
    </solutionobject>
  </level>
</flatModel4Unity:Game>
```

Fig. 14. XML-based version of the generated scenario.

A fictive child's profile was used to drive the focus on a targeted skill and therefore experimenting all difficulty levels mappings, one-by-one. These experiments with experts highlighted some misconceptions about some min/max ranges for randomized instantiations. Some generated scenes were too complex to solve because the randomize algorithm chose the max number of possible objects for a scenes declaring a large number of objects locations. Ranges have been then reviewed to fixed values and that are not dependent of other elements.

6 Conclusion

This chapter focuses on the development of a learning game for helping young children with Autistic Syndrome Disorder to learn and generalize visual performance skills. It tackles the issue of generating adapted learning game scenarios

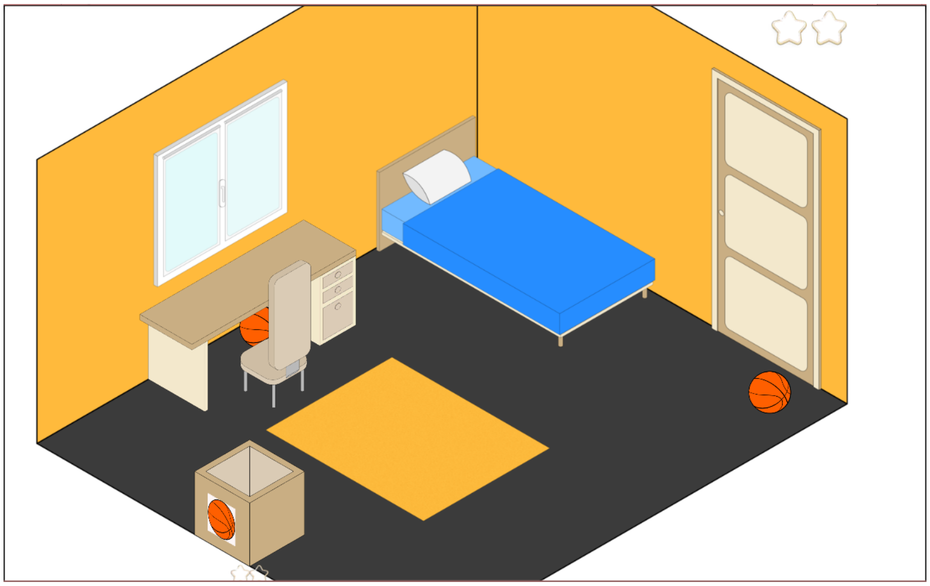

Fig. 15. The B4/bedroom scene set-up according to the data from the generated XML-based scenario.

by proposing a Model-Driven Engineering approach. The proposal 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. The approach 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.

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. These dynamical domain rules being doomed to evolve by the expert after validation sessions, we are working to make them explicit with a view to manipulating these rules as additional inputs of the leaning scenarios generator.

References

1. Burton, L.R., McEachin, J.: A Work in Progress: Behavior Management Strategies and a Curriculum for Intensive Behavioral Treatment of Autism. New York: DRL Books (1999)
2. Callies, S., Sola, N., Beaudry, E., Basque, J.: 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). pp. 107–116 (2015)
3. Deterding, S., Dixon, D., Khaled, R., Nacke, L.: From game design elements to gamefulness: Defining "gamification". In: Proceedings of the 15th

International Academic MindTrek Conference: Envisioning Future Media Environments. pp. 9–15. MindTrek '11, ACM, New York, NY, USA (2011). https://doi.org/10.1145/2181037.2181040
4. Ern, A.: The use of gamification and serious games within interventions for children with autism spectrum disorder (January 2014), http://essay.utwente.nl/64780/
5. van Est, C., Bidarra, R.: High-level scenario editing for simulation games. In: Proceedings of the 6th International Conference on Computer Graphics Theory and Applications. (2011)
6. Hussaan, A.M., Sehaba, Karim, e.K., Sharples, M., Klobučar, T.: "consistency verification of learner profiles in adaptive serious games". In: 11th European Conference on Technology Enhanced Learning. pp. 384–389. Springer International Publishing (2016)
7. Janssens, O., Samyn, K., Van de Walle, R., Van Hoecke, S.: 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) (06 2014)
8. Kolovos, D.S., Paige, R.F., Polack, F.A.C.: The epsilon transformation language. In: Vallecillo, A., Gray, J., Pierantonio, A. (eds.) Theory and Practice of Model Transformations. pp. 46–60. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
9. Laforcade, P., Loiseau, E., Kacem, R.: A Model-Driven Engineering Process to Support the Adaptive Generation of Learning Game Scenarios. In: CSEDU. vol. 1. INSTICC, Funchal, Madeira, Portugal (Mar 2018), https://hal.archives-ouvertes.fr/hal-01740062
10. Laforcade, P., Vakhrina, V.: 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 (2015)
11. Loiseau, E., Laforcade, P., Mawas, N.E.: 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. pp. 64–76 (2015). https://doi.org/10.5220/0005434000640076
12. Lopes, R., Bidarra, R.: Adaptivity challenges in games and simulations: A survey. IEEE Transactions on Computational Intelligence and AI in Games **3**(2), 85–99 (June 2011). https://doi.org/10.1109/TCIAIG.2011.2152841
13. M. Bielikova, M. Diveky, P.J.R.K., Omelina, L.: Automatic generation of adaptive, educational and multimedia computer games. vol. 2, pp. 371–384. Signal Image Video Process. (2008)
14. 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., Whittle, J.: The Relevance of Model-Driven Engineering Thirty Years from Now. In: Dingel, J., Schulte, W., Ramos, I., Abraho, S., Insfran, E. (eds.) Model-Driven Engineering Languages and Systems. Model-Driven Engineering Languages and Systems, vol. 8767, p. 18. Springer International Publishing Switzerland, Valencia, Spain (Sep 2014). https://doi.org/10.1007/978-3-319-11653-2_12, https://hal.inria.fr/hal-01081848
15. Partington, J., Analysts, P.B.: The Assessment of Basic Language and Learning Skills-revised (the ABLLS-R) (2010)
16. Sehaba, K., Hussaan, A.: Goals: generator of adaptive learning scenarios. In: International Journal of Learning Technology. vol. 8, pp. 224–245 (10 2013)

17. Sina, S., Rosenfeld, A., Kraus, S.: Generating content for scenario-based serious-games using crowdsourcing. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. pp. 522–529. AAAI Press (2014)
18. Steinberg, D., Budinsky, F., Paternostro, M., Merks, E.: EMF: Eclipse Modeling Framework 2.0. Addison-Wesley Professional, 2nd edn. (2009)
19. Whyte, E.M., Smyth, J.M., Scherf, K.S.: Designing serious game interventions for individuals with autism. Journal of Autism and Developmental Disorders **45**(12), 3820–3831 (Dec 2015). https://doi.org/10.1007/s10803-014-2333-1, https://doi.org/10.1007/s10803-014-2333-1
20. Zakari, H.M., Ma, M., Simmons, D.: A review of serious games for children with autism spectrum disorders (asd). In: International Conference on Serious Games Development and Applications. pp. 93–106. "Springer International Publishing" (2014)