

Towards Model-Driven Requirements Engineering for Serious Educational Games: Informal, Semi-formal, and Formal Models

Kendra M.L. Cooper^{1,*}, Eman S. Nasr², C. Shaun Longstreet³

¹The University of Texas at Dallas, Richardson, U.S.A.
kendra.m.cooper@gmail.com

²Independent Researcher, Cairo, Egypt
nasr.eman.s@gmail.com

³Marquette University, Wisconsin, U.S.A.
christopher.longstreet@marquette.edu

Abstract. Serious educational games (SEGs) are receiving significant attention, as they provide immersive, engaging learning environments with a rigorous pedagogical foundation. SEG engineering requires an interdisciplinary approach involving game developers, educators, and software engineers. The requirements engineering (RE) community has substantial expertise in processes, notations, tools, and techniques. Here, we explore how can we tailor and adopt this expertise for developing SEGs with a three step model-based approach that integrates established techniques: create an informal model of the SEG requirements (narrative captured like a storyboard); transform the narrative into a semi-formal, tailored UML use case model (visual and tabular, using templates); transform the semi-formal model into formal models for testing and verification. A collection of SEGs (test games) has been created using the process; currently the transformations are performed manually. The formal model is represented in XML, which can be loaded, played, and tested in the game engine. In the future, we will explore semi-automatically transforming the models and creating Statechart models, which can be verified using simulations.

Keywords: requirements engineering, serious educational games, model-driven.

1 Introduction

Serious educational games (SEGs) have significant pedagogical potential as they provide immersive, engaging and fun environments that require deep thinking and complex problem solving within a construct of overcoming obstacles and challenges [1]. They create interactive student-centered environments rather than a passive content-centered classroom environment. SEGs are complex applications, requiring expertise from multiple disciplines including game development, education, and software engineering. Established game development approaches are document-centric, often using a pre-production game design document and production game software

requirements specification [2]. Recently, the potential value using model-driven engineering (MDE) [3] approaches for games [4] and more specifically to SEGs [5] has been presented. UML-based game specifications have also been considered that focus on tailoring Statecharts [6]; they offer a rigorous, state machine foundation, but may be more difficult for some stakeholders (e.g., game designers) to use. Game designers with experience using Storyboards [7], for example, may find a tabular use case (UC) specification format quite familiar, as Storyboards are often presented as a sequence of cells in a tabular format. Tabular specifications are well-established [8]; they are considered straightforward to define, understand, and maintain.

To improve the development of SEGs, our research project, SimSYS, is underway. SimSYS is a MDE approach that uniquely integrates elements of traditional game design, pedagogical content, and software engineering methodologies. Our proposed approach has three main steps: create an informal model of the SEG requirements (captured like a storyboard with textual descriptions of the learning objectives and game play in addition to user interface concepts e.g., graphics, audio); transform the informal model into a semi-formal tailored UML UC model (visual and tabular, template based specifications) [9]; transform the semi-formal model into formal, executable models (Statechart for comprehensive simulation and XML [10], which can be loaded and played in the game engine). Our approach adopts and tailors well-established solutions (e.g., Storyboards, UML UC, XML, Statecharts) as the need for a completely new solution is not evident at this time. Our MDE-based approach can be applied in an agile, iterative development process, for example, by describing a part of the game and allowing earlier assessment and feedback. Learning objectives, for both topic specific subject matter and transferable skills, are thoroughly integrated.

In previous work, we have presented an underlying meta-model for SimSYS [11] and preliminary work on the models, with a focus on the semi-formal, tabular templates [12]. The meta-model facilitates the development of high-quality, engaging, educational games because it explicitly ties knowledge requirements, transferable skills and course outcomes to game production. The templates, which embody the meta-model concepts, help to systematically and efficiently structure SimSYS games into Acts, Scenes, Screens, and Challenges.

Here, we present initial results on our proposed three step approach to create and refine informal, semi-formal, and formal (XML, Statechart) models. The validation uses an internally defined collection of simple test games, which currently has six symbolic games, an algebra game for grade four students, and a software engineering game on design patterns. The approach is illustrated with part of a Challenge from the software engineering test game; we focus on the transformation to an XML representation that can be loaded and tested in a game engine. An overview of the new approach is presented in Section 2. The manual transformations (informal to semi-formal and semi-formal to formal) are introduced in Sections 3 and 4. Conclusions and future work are in Section 5.

2 Overview

The SimSYS approach to engineering games has three main steps (Figure 1). The first step is to create an informal, high level model of the SEG as a narrative. The narrative captures the game like a Storyboard with textual descriptions of the learning objectives and game play in addition to user interface concepts (e.g., graphics, audio). This model is important as it allows the game designers to focus on the creative aspects of the game play, in place of a pre-production game design document.

The second step is to transform the informal model into a semi-formal tailored UML UC model (visual and tabular, template-based specifications). UML UCs have been adopted as they are a well-known notation that can be tailored. The overall game is organized into Acts, Scenes, Screens, and Challenges. Each of these has a tabular template to assist in the game development. Tabular, template-based representations are considered straightforward to develop, review, and maintain; they allow the modularization of a specification into sub-tables to manage complexity. As this semi-formal model is developed and reviewed, errors can be identified and corrected.

The third step is to transform the semi-formal model into formal models (Statechart and XML). The Statechart model can undergo comprehensive simulation/animation to verify its behavior using commercial tool support. The XML model is the game specification, which can be loaded, played, and tested using the SimSYS Game Play Engine. The errors identified using testing or simulation/animation techniques can be rapidly corrected, across the formal and semi-formal models.

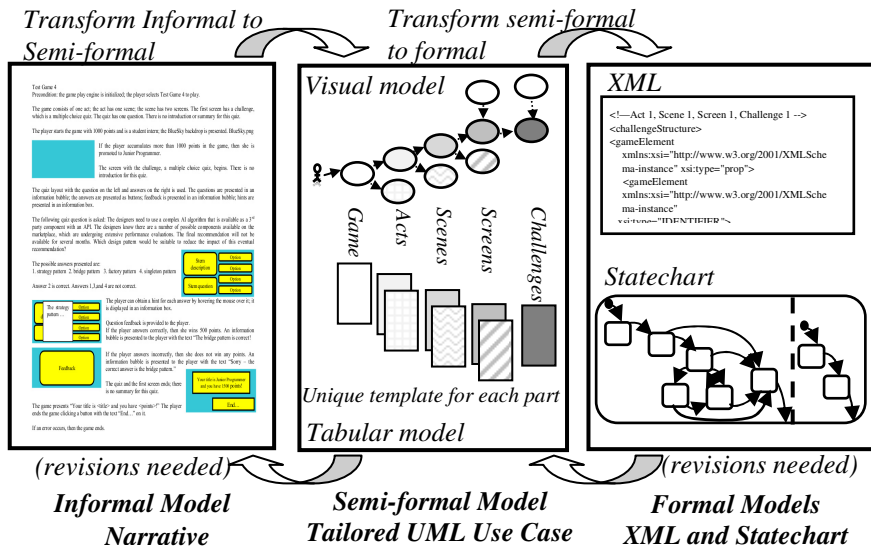


Fig. 1. Overview of Models in the SimSYS Approach (informal, semi-formal, formal)

3 Transforming the Informal to Semi-formal Model

The informal model is organized into a title, overview of the game (number of acts, scenes, screens, challenges), the learning objectives (both topic specific and transferable skills), initial conditions for the game (such as number of points a player starts with, backdrop for the scenes), and rules that need to be applied as the game is played. After this, the game play is described as a sequence of game play interactions over time, followed by the conditions at the end of the game (Figure 2). The game play interactions include Challenges for the player that address specific learning objectives.


Title: Test Game 4 SE Design	
Overview: The game consists of one act; the act has one scene; the scene has two screens. The first screen has a challenge, which is a multiple choice quiz. The quiz has one question on design patterns; it is a dialogue question requiring critical thinking, problem solving, and analysis skills in addition to SWEBOK Software Design topics (general, process, context). The Bloom's taxonomy categories are knowledge and application. There is no introduction or summary for this quiz. The second screen provides a summary of the player's progress in the game.	
	Initial conditions: The player starts the game with 1000 points and is a student intern; the BlueSky backdrop is presented (BlueSky.png).
Game rule (how to win): If the player accumulates more than 1000 points...	

Fig. 2. Informal Model of a (Partial) Test Game

The narrative is manually, iteratively refined into a tailored UC model (visual, textual). The approach has a stereotyped UC for the Game, Act, Scene, Screen, and Challenge; each has a corresponding tabular template [12]. The Game captures the learning objectives for the game, the one or more acts it is organized into with their transitions, and the player and non-player characters. The Act captures the learning objectives addressed and the one or more scenes each one is organized into; the transitions from one scene to another in the act are defined. The Scene captures the learning objectives addressed; the one or more screens each one is organized into with their transitions, in addition to the backdrop and background music. The Screen and Challenge capture the detailed game play. The Screen UC captures the learning objectives addressed, the characters and props involved, with their placement, animation, sound effects. Props include generic interaction elements (e.g., information boxes, conversation bubbles, information bubbles), domain specific elements (e.g., whiteboard, blackboard), and set decorations (e.g., furniture, coffee cup). Background material for the storyline or educational material could be included in the game by presenting it on a whiteboard or as a conversation bubble for a character. A screen may have an optional challenge, which could be a dialog based quiz (untimed, timed, competition, non-competitive) or a problem solving exercise in a sandbox. Part of a Challenge in the semi-formal representation is illustrated in Figure 3 (left-hand side).

4 Transforming Semi-formal to XML Formal Model

The collection of tabular, semi-formal representations of the Game, Acts, Scenes, Screens, and Challenges is transformed into a single XML file that can be loaded into

the SimSYS Game Play Engine. The manual transformation is done one part at a time, using tags in a schema definition to organize the content. There are tags, for example, to organize the overall model with respect to the Game, Acts, Scenes, Screens, and Challenges. The tags are nested, to reflect the composition relationships (a Game has one or more Acts, an Act has one or more Scenes, a Scene has one or more Screens, a Screen has an optional Challenge). Part of the XML file is illustrated in Figure 3 (right-hand side), focusing on the Challenge. Within a Challenge, for example, one question in the quiz is related to subject specific topics in SE Design and transferable skills described in the tabular specification (What kind of knowledge is in the challenge?); these are represented in XML using two kinds of tags to define lists of topics (<domainKnowledgeList>, <transferableKnowledgeList>). Following this (not presented in the Figure), the assessment approach (e.g., Bloom taxonomy categories) and one or more quiz question with answer options, evaluation, hints, rewards, and feedback for the player are defined. As the mapping proceeds, errors are detected and corrected in the semi-formal and informal models, making the approach highly iterative. The manual mapping from the semi-formal tabular model to the XML model is time consuming, but not too difficult.

Test Game on SE Design Patterns - Act 1, Scene 1, Screen 1, Challenge 1

Identifier	Challenge 1	<Challenge>
Purpose	Design pattern selection. Challenge type: Multiple choice quiz (dialogue)	<id>Challenge 1</id> <scope>design pattern selection</scope> <type>Multiple choice quiz</type>...
Learning Objectives	Domain Specific Skills Software Engineering, Software Design Standard: SWEBOOK 2004 1. Software Design Fundamentals 1.1 General Design Concepts 1.2 Context of Software Design 1.3 Software Design Process 1.3.2 Detailed Design 3. Software Structure and Architecture 3.2 Design Patterns Transferable Skills Analysis Critical Thinking Problem Solving Bloom's Taxonomy: knowledge, application Understand the purpose of established design patterns ...	<! What kind of knowledge is in the challenge? --> <domainKnowledgeList> <domainKnowledge>software engineering </domainKnowledge> <standard>SWEBOOK 2004</standard> <area>Software Design</area> <subarea>Software Design Fundamentals</subarea> <topic_list> <topic>General_Design_Concepts</topic> <topic>Context_of_Software_Design</topic> <topic>Software_Design_Process</topic> <subtopic>Detailed Design</subtopic> <subarea>Software Structure and Architectures</subarea> <topic>Design Patterns</topic> </topic_list> </domainKnowledgeList> <transferableKnowledgeList> <transferableKnowledge>analysis </transferableKnowledge> <transferableKnowledge>critical_thinking </transferableKnowledge> <transferableKnowledge>problem_solving </transferableKnowledge> </transferableKnowledgeList>

Fig. 3. Semi-formal Model and Formal Model for a (partial) Test Game

5 Conclusions and Future Work

Initial results on an iterative, three step, model-based RE approach for SEGs is previewed here. The approach begins with an informal narrative, like a Storyboard, which is subsequently transformed into a tailored semi-formal UML UC model (with visual and tabular specifications) and lastly into a formal XML model. Learning objectives, for example, are initially captured in natural language in the informal

narrative; transformed into the semi-formal model in a learning objective row, then represented with nested XML tags in the formal model. The XML model can be loaded and tested in the SimSYS Game Play Engine. In applying the three step approach to engineering a collection of test games, we have found it to be straightforward, but labor intensive. In particular, the formal XML model required significant time to create. An Intelligent Semi-Automated Game Generation module is currently being prototyped to alleviate the effort in populating the informal narrative and semi-formal models; a goal is to automatically generate the formal models. Related work is available on automating UC transformations, indicating this is feasible [13]. In the future, additional validation is needed, both by continuing our own project and by external educational game researchers and educators. The transformation from the semi-formal model to the Statechart model is also planned to explore model verification via simulation/animation using commercial tool support.

References

1. Gee, J.P.: What video games have to teach us about learning and literacy. Macmillan, U.S.A. (2003)
2. Adams, E.: Fundamentals of Game Design, 2nd edn. New Riders Publishing (2010)
3. Object Management Group, OMG Model Driven Architecture (MDA) Guide Version 1.0.1 (2003), <http://www.omg.org>
4. Dormans, J.: The Effectiveness and Efficiency of Model Driven Game Design. In: Herrlich, M., Malaka, R., Masuch, M. (eds.) ICEC 2012. LNCS, vol. 7522, pp. 542–548. Springer, Heidelberg (2012)
5. Tang, S., Hanneghan, M.: Fusing Games Technology and Pedagogy for Games-Based Learning Through a Model Driven Approach. In: Proceedings of the 2011 IEEE Colloquium on Humanities, Science, and Engineering Research, pp. 380–385 (2011)
6. Sauer, S., Engels, G.: UML-based Behavior Specification of Interactive Multimedia Applications. In: Proceedings of the IEEE 2001 Symposia on Human Centric Computing Languages and Environments, pp. 248–255 (2001)
7. Truong, K., Hayes, G., Abowd, G.: Storyboarding: an empirical determination of best practices and effective guidelines. In: Proceedings of the 6th Conference on Designing Interactive Systems, pp. 12–21 (2006)
8. Pollack, S.L., Hicks, H.T., Harrison, W.J.: Decision tables: theory and practice. Wiley-Interscience (1971)
9. Object Management Group, OMG Unified Modelling Language, version 2.2 (2009), <http://www.omg.org>
10. World Wide World Consortium, Extensible Markup Language (XML) 1.0, 4th edn. (August 2006), <http://www.w3.org/TR/xml/>
11. Longstreet, C., Cooper, K.: A meta-model for developing simulation games in higher education and professional development training. In: Proceedings of the IEEE 17th International Conference on Computer Games, pp. 39–44 (2012)
12. Cooper, K., Longstreet, C.: Towards Model-driven Game Engineering for Serious Educational Games: Tailored Use Cases for Game Requirements. In: Proceedings of the IEEE 17th International Conference on Computer Games, pp. 208–212 (2012)
13. Riebisch, M., Hübner, M.: Refinement and Formalization of Semi-Formal Use Case Descriptions. In: Proceedings on the 2nd Workshop on Model-Based Development of Computer Based Systems: Appropriateness, Consistency and Integration of Models (2004)