Gamification of Software Modelling Learning

Alfa Yohannis, Dimitris Kolovos, and Fiona Polack

Department of Computer Science, University of York, York, United Kingdom
ary506@york.ac.uk

Abstract. Software modelling has a fundamental role in software engineering. However, it is perceived as relatively difficult for learners to develop the necessary abstraction skills to master the subject. Gamification is now flourishing as a popular strategy to engage learners. This research attempts to exploit gameful design as an innovative approach, used to create games that reinforce learners' mastery of software modelling by developing their abstraction skills. Our approach to gameful design brings together gamification development concepts such as design lenses and intrinsic skill atoms, and pedagogical design principles from several learning theories and models. The research follows the Design Science Research Methodology and exploits Model-Driven Engineering best practices. To date, the research has been developing an early prototype of gamification for learning software modelling. For evaluation, the effects of the artefact will be measured using longitudinal controlled experiments.

Keywords: software modelling, gamification, learning, abstraction

1 Introduction

Software modelling is commonly perceived as a difficult subject since it requires a mastery of abstraction [1]. However, this subject has a fundamental and crucial role in software engineering education and practice. Failure to master this topic will affect the students abstraction capability in analysing and designing a real-world software. Weak software modelling will likely cause software engineering students to face difficulties completing their degrees, as most of the software engineering related subjects have a sense of intrinsic abstraction problems [2]. Students' perception of software modelling will affect their attitude towards software engineering today and their career paths in the future.

The problems of learning appropriate abstraction skills for software modelling is similar to problems in mathematics, where most of the concepts can only be accessed through symbolical representations [3]. Abstraction also requires students to perform information hiding, generalisation, approximation or reformulation, leaving out the irrelevant aspects but keeping the relevant ones, or separation from the concrete reality [4]. To overcome these challenges, we need to put more effort into software modelling learning design, developing a more concrete and motivating presentation which can engage students and facilitate deep learning.

In recent years, the use of games or game elements for ~~serious~~ purposes other than leisure has drawn lots of attention. Gamification [5] and Serious Games [6] have been viewed as solutions to motivational problems that emerge when users are required to engage in activites that they perceive as boring, irrelevant or difficult, e.g. Learning sorting algorithms [7] or C-programming [8].

Therefore, the purpose of this research is to investigate and develop a gamification design framework that systematically and semi-automatically drives gamification design to produce better designed software modelling games. More precisely, this research aims to answer the following research questions ~~that are derived from the purpose of this study~~:

1. Which processes, aspects, principles, or components of software modelling and their teaching and learning practices should be included?
2. What types of game elements and their roles that can deliver software modelling learning best?
3. What kind of orchestrating framework is needed to design the interaction between, software modelling and game elements to produce a better software modelling gamification?
4. To what extent does gamification of software modelling improve engagement and motivation and improve learners performance?

Due to the limitation of number of pages, this paper does not cover ~~some interesting~~ aspects of this research, such as the architecture of the artefact and the validation methods applied to evaluate models created by learners.

2 Related Work

Several approaches attempt to bring software modelling into a more concrete presentation that can be easily understood by learners, ranging from didactic learning [9], modelling tools utilization [10], alternative communication channels and the use of modelling language [11], immersive visual modelling through virtual environment [12], software design studio [13], project-based approach [14], to code generation investigation [15]. However, most of the approaches have weaknesses in motivating learners to engage continuously, frequently, and actively to learn software modelling, which is the important aspect impacting greatly on learning [16]. To address lack of engagement, we investigate game-based learning, to learn or teach software modelling. This method provides students with a new way of learning software modelling, which is not only interactive but also engaging enough to keep them learning continuously.

The use of game elements for a purpose other than leisure is called gamification [5]. Gamification design is still an ongoing challenge [17], and, to date, there is no gamification design framework that particularly structures the design of software modelling gamification~~; a framework that integrates game specific domain into software modelling. Hence, this research aims to develop a gamification design framework of software modelling learning.~~

Most of the software related gamification studies available are related to software engineering in a larger context or to other aspects of software engineering, such as software implementation and project management~~, rather than software modelling in particular~~ [18]. After the literature exploration, only four works have been identified applied gamification for software modelling. None of these works addresses software modelling learning in general~~ – how learners can learn abstraction in modelling~~. Instead, they address specific topics such as activity diagramming [19], coupling and cohesion [20], and enterprise architectures [21] [22]. Most of the works also cover pedagogical aspect superficially or not at all and validation is restricted to a very limited number of users [19][20][21].

3 Research Methods

Since the output of this work is designed artefacts, we decided to utilise the Design Science Research Methodology (DSRM) [23] as our umbrella methodology. DSRM is selected because it provides a comprehensive high-level conceptual framework how to undergo a full-cycle research process. It also provides six activity guidelines for understanding, developing, executing, and evaluating design artefacts. The six activities are (1) problem identification and motivation ~~activity~~, (2) definition of objectives for a solution ~~activity~~, (3) setting of targets for a solution ~~activity~~, (4) design and development ~~activity~~, (5) demonstration and evaluation ~~activities~~, and (6) communication.

The high-level characteristics of DSRM mean that we can employ other research methods as the sub-methods in each activity. For example, we employ interviews, literature reviews, and discussion with experts as our methods in problem identification and motivation activity, as well as using Deterding's lens of intrinsic skill atoms [24] to produce a gameful design in the design and development activity.

4 Gamification Design: Design Lenses

Deterding et al. [24] define various design lenses, to focus analysis of the game requirements. Here, we only describe two of all the lenses applied in our gamification design. The resulting artefact is displayed in Figure 1.

Challenge Lens. The design of the levels of the game has a gradually increased difficulty as well as variety in its challenges, to expose learners to different kinds of domains, models, and diagrams. The onboarding game element is also planned to be implemented into the artefact to help learners familiar with the control system and the flow of the game. The software modelling game design will also utilise templates to help learners building models without having to start from scratch. ~~The foundation model is already given. They only need to continue the model the meets certain objectives.~~

Action and Object Lens. A real world problem can be very exhaustive and time-consuming. Thus, the extraneous activities that are not relevant to the core concepts that are being taught should be removed. As a result, learners

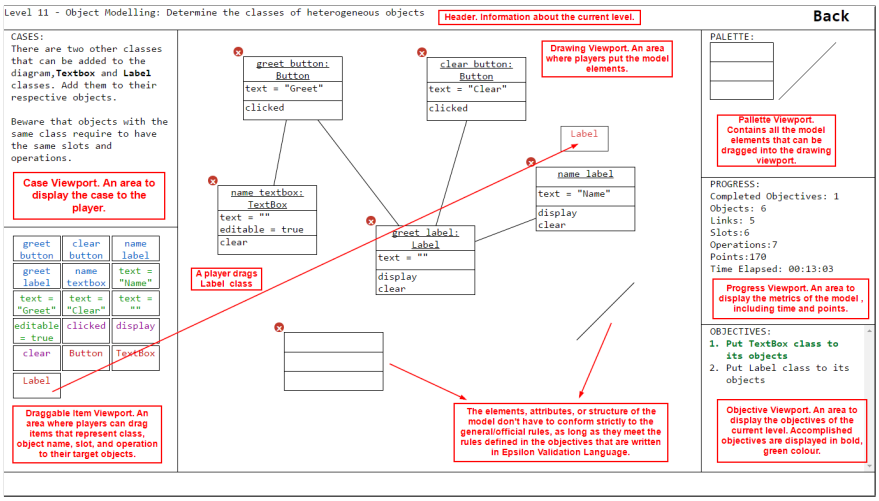

Fig. 1: The game's display.

will be more focused on the main concepts. For that reason, game elements like bite-sized actions (e.g. drag and drop), limited choices (i.e. only limited items can be dragged), and microflow (i.e. put the right element to its right place) are selected to facilitate learners in performing the core activities. Likewise, an element of underdetermination is used to provoke learners' creativity since most of the time there is no single correct model to represent a domain; the models can vary. Attractive design is also significant since it draws learners' attention to interact with the artefact.

5 Gamfication Design: Intrinsic Skill Atoms

The design of the cycle of the game mechanics is derived from Intrinsic Skill Atoms [24], which have six components that should be defined: motivation, goals, actions and objects, challenges, rules, and feedbacks. The main motivation for engaging with our game artefact is to master software modelling, which means that they will be able to construct models and solve model-related problems. For instance, students should be able to build an object model for a sign-in screen. Moreover, The goal of their modelling activities is to meet the given objectives whilst obeying the rules. For example, learners are asked to construct an object diagram of a button that when pressed will clear the text of a textbox; there will be two objects (a button object and a textbox object), but the diagram will not be complete if there is no link that connects them to represent their relationship.

As in most games, learners are required to take some actions to manipulate some objects in order to achieve the modelling goals. The artefact requires learners to perform dragging and dropping and connecting diagram elements (e.g. objects and links) to construct a model. There are also draggable items—

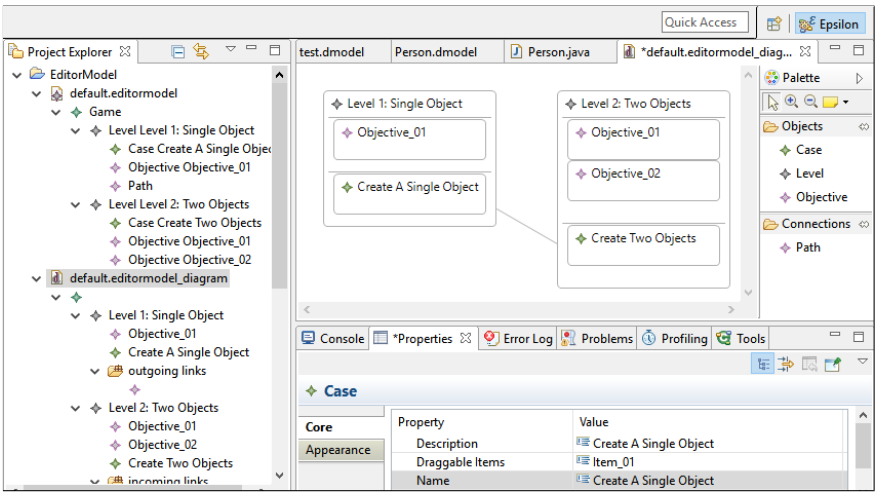

Fig. 2: The visual modelling editor to automatically generate the game.

containing the names or values of slots, actions, and classes of objects—that must be associated to the appropriate diagram elements. The learners are confronted with some challenges in the form of problem-solving cases to make the modelling activity more exciting. For example, an animation that shows how a login page works is displayed to the learners, and they are asked to model it.

6 Visual Modelling Editor

To support developers in the design and customisation of gamification for software modelling learning—the integration of game elements into its learning activities—at a high level of abstraction and so as to automatically build the game generator, this research is developing a visual modelling editor (Figure 2) using Eugenia, a GMF-based graphical model editors [25]. Currently, developers can use the editor to design the ~~gamification of software modelling learning by defining its~~ flows, levels, challenges, and objectives. In the future, this research plans to add more features, such as user-customised types of modelling and diagrams.

7 Evaluation

The evaluation of the game artefact will use controlled experiments. The respondents, software modelling students, will be divided into two groups, a control group and experimental group. The control group will learn software modelling using traditional methods while the experimental group will learn with support from the artefact. After some time learning, their performance will be measured by their ability to solve the problems. To evaluate the generality of the effects

of the artefact, a longitudinal experiment is also considered as well as undertaking the experiments in different countries and universities (the evaluation of the gamification design framework is similar to the game artefact's evaluation except for the respondents and problems will be software developers and software modelling gamification construction).

Additionally, surveying with questionnaires or interviews might be conducted to investigate the underlying variables or processes. Structural equation modelling [26] is also an option if measuring the effects of the identified underlying variables is required. An alternative method for understanding of the underlying variables and processes is through investigating the artefact's event logs using data mining or machine learning techniques.

8 Conclusion

This paper explains our research motivation and problem statements, proposed solution and objectives, research methods, the current progress of the design and development of the artefact, and the evaluation plan. So far, this research is focusing its work on software modelling learning. In the future, this research plans to address metamodelling and model transformation learning.

Acknowledgments. Thanks to York Masters students who participated in our preliminary surveys. This research is supported by *Lembaga Pengelola Dana Pendidikan Indonesia* (Indonesia Endowment Fund for Education).

References

1. J. Börstler, L. Kuzniarz, C. Alphonce, W. B. Sanders, and M. Smialek, "Teaching software modeling in computing curricula," in *Proceedings of the final reports on Innovation and technology in computer science education 2012 working groups*, pp. 39–50, ACM, 2012.
2. J. Kramer, "Is abstraction the key to computing?," *Communications of the ACM*, vol. 50, no. 4, pp. 36–42, 2007.
3. R. Duval, "A cognitive analysis of problems of comprehension in a learning of mathematics," *Educational studies in mathematics*, vol. 61, no. 1-2, pp. 103–131, 2006.
4. L. Saitta and J.-D. Zucker, *Abstraction in artificial intelligence and complex systems*, vol. 456. Springer, 2013.
5. S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining gamification," in *Proceedings of the 15th international academic MindTrek conference*, pp. 9–15, ACM, 2011.
6. D. R. Michael and S. L. Chen, *Serious games: Games that educate, train, and inform*. Muska & Lipman/Premier-Trade, 2005.
7. A. Yohannis and Y. Prabowo, "Sort attack: Visualization and gamification of sorting algorithm learning," in *the 7th VS-Games*, pp. 1–8, IEEE, 2015.
8. M.-B. Ibanez, A. Di-Serio, and C. Delgado-Kloos, "Gamification for engaging computer science students in learning activities: A case study," *IEEE Transactions on Learning Technologies*, vol. 7, no. 3, pp. 291–301, 2014.