Integrating Usability Evaluation into Model-Driven Video Game Development

Adrian Fernandez, Emilio Insfran, Silvia Abrahão, José Carsí, Emanuel Montero

Integrating Usability Evaluation into Model-Driven Video Game Development

Adrian Fernandez, Emilio Insfran, Silvia Abrahão
José Ángel Carsí and Emanuel Montero

ISSI Research Group, Department of Information Systems and Computation
Universitat Politècnica de València
Camí de Vera s/n, 46022, Valencia, Spain
{afernandez, einsfran, sabrahao, pcarsi, emontero}@dsic.upv.es

Abstract. The increasing complexity of video game development highlights the need of design and evaluation methods for enhancing quality and reducing time and cost. In this context, Model-Driven Development approaches seem to be very promising since a video game can be obtained by transforming platform-independent models into platform-specific models that can be in turn transformed into code. Although this approach is started to being used for video game development, there is a need for usability evaluation methods specifically tailored to this type of development process. In this paper, we present a usability inspection method that can be used along all the stages of the model-driven video game development. The method relies on a Usability Model that is aligned with the ISO/IEC 25010 (SQuaRE) standard and decomposes usability into measurable attributes and metrics specific for the video game domain.

Keywords: Video Game, Usability Inspection, Model-Driven Development.

1 Introduction

The video game development industry is a strong economic sector that deals with the development of highly interactive software, i.e., video games, for a wide variety of technology platforms such as PCs, consoles, Web browsers, and mobile devices. The interaction between the game and the players is a critical factor in the success of a video game. Usability and playability are considered to be the most important quality factors of video games [13]. *Usability* is defined as the degree to which the video game can be understood, learned, used and is attractive to the user, when used under specified conditions [10]. *Playability* is defined as a collection of criteria with which to evaluate a product's gameplay or interaction [11]. Playability is often evaluated by using early prototypes and iterative cycles of playtesting during the entire video game development cycle. However, the evaluation of usability in current video game development practices is often deferred to late stages in the game development cycle, thus signifying that usability problems from early stages may be propagated to late stages of the development, and consequently making their detection and correction a very expensive task.

A model-driven video game development approach could provide a suitable context for rapid iteration early in the development cycle. Platform-independent (or

platform-specific) models (i.e., PIM or PSM) can be evaluated during the early stages of video game development to identify and correct some of the usability problems prior to the generation of the source code of the final video game application. We are aware that not all the usability problems can be detected based on the evaluation of models since they are limited by their own expressiveness and, most important, they may not predict the user behavior and preferences. However, as suggested by previous studies [4], the use of inspection methods for detecting usability problems in product design (models in our context) can be complemented with other evaluations performed with end-users before releasing a video game to the public.

The contribution of this paper is a usability inspection method that can be integrated in early stages of model-driven video game development. This method relies on a Video Game Usability Model which decomposes the usability characteristic proposed in the ISO/IEC 25010 (SQuaRE) standard [10] with new usability attributes for the video game domain. These attributes are quantified through their association with generic measures that can be operationalized by establishing a mapping between their generic definition and the specific modeling primitives of the software artifacts to be evaluated.

This paper is organized as follows. Section 2 discusses usability evaluation techniques for video game development. Section 3 discusses our strategy for integrating usability into model-driven video game development. Section 4 describes the Video Game Usability Model while Section 5 proposes a strategy to apply this model for performing early usability evaluations in model-driven video game development. Finally, Section 6 presents our conclusions and further work.

2 Related Work

The state of the art for game development in Software Engineering has been recently summarized in a systematic literature review [3]. The results of this review show a significant lack of studies in the key dimensions of video game quality: playability and usability. However, some efforts have been made to integrate current usability evaluation methods into the game development industry and game research, and a brief review of current game usability techniques has been provided in [13].

Some usability evaluation methods (usually referred as empirical methods) are based on capturing and analyzing usage data from real players. Some representative examples are *think-aloud* methods and *focus group* [9]. In *think-aloud* methods, the player sits down to play the video game and narrates his experiences while a user experience evaluator sits nearby listening and taking notes. In *focus group* methods, game designers gather a small group of potential game players together to discuss their opinions of the design of the interface, along with the game mechanics and story.

Other kind of usability evaluation methods (usually referred as inspection or analytical methods) are performed by expert evaluators or game designers and are based on reviewing the usability aspects of software artifacts (which are commonly game user interfaces) with regard to their conformance with a set of guidelines. The most representative example is *heuristic evaluation,* which is a common inspection method for evaluating the usability of video game interfaces in both early and functional game prototypes. Examples of heuristic evaluation methods were presented in the work of Federoff [7] and Pinelle et al. [15], in which a set of guidelines for

creating a good game were defined, based on the experience of a game development case study, and PC game reviews, respectively.

In this paper, we focus on inspection methods since they do not involve the players' participation and can be employed during the early stages of the game development process. Specifically, our method extends previous approaches by providing specific video game usability attributes and measures that can be quantified by means of model-transformations. The objective is to reduce the subjectivity of existing methods that are mainly based on plain checklists of desired features with no specific guidelines on how they can be applied. Model-driven development provides a suitable context for early usability evaluations since traceability between high-level artifacts (models) and source code is maintained throughout the development process [1]. Finally, approaches based on usability models have been successfully employed as inspection methods in other domains, such as model-driven software development [2] and model-driven Web development [8]. However, as far as we know, no usability model has been proposed for model-driven video game development.

3 Usability in Model-Driven Video Game Development

The usability of a video game application obtained as a result of a transformation process can be assessed at several stages of a model-driven development process. We propose the use of a Video Game Usability Model which contains a set of usability attributes and measures that can be applied by the video game designer in the following phases of a MDA-based development process: **i)** in the PIM, to assess different models that specify the video game application independently of platform details (e.g., screen flow diagrams, screen mock-ups, screen navigation diagrams); **ii)** in the PSM, to assess the concrete design models related to a specific platform; and **iii)** in the code model, to assess the generated video game application (see Fig. 1).

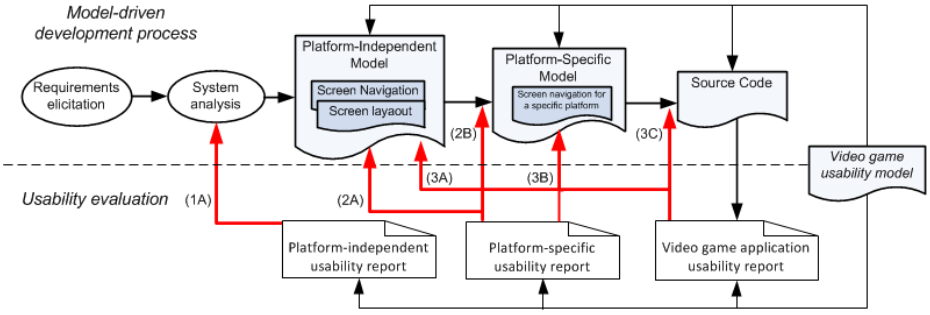

Fig. 1 Integrating a Video Game Usability Model into model-driven development processes

It should be noted that the process is driven by the PIM, which is automatically transformed into a PSM, and this PSM into source code. Therefore, the evaluations performed at the PIM produce a *platform-independent usability report* that provides a list of usability problems with recommendations to improve the system analysis stage (Fig. 1 (1A)). Changes in the PIM are reflected in the CM by means of model transformations and explicit traceability between models. This prevents usability problems to appear in the generated video game application (CM).

The video game designer should select the set of relevant usability attributes and measures from the Video Game Usability Model. There are some usability attributes (e.g., degree of attractiveness) that can only be evaluated on a specific platform and taking into account the specific components of the video game UI (PSM) or the components that build the final application (CM). Evaluations performed at the PSM produce a *platform-specific usability report*. If the PSM does not allow obtaining an application with the required level of usability, the report will suggest changes to correct the following: the PIM (Fig. 1 (2A)), the transformation rules that transform the PIM into PSM (Fig. 1 (2B), and/or the PSM itself (Fig. 1 (3B)). Nevertheless, the evaluations at the PIM or PSM level should be done in an iterative way until these models allow generating a video game application with the required level of usability.

Finally, evaluations performed at the CM level produce a *final application usability report*. Rather than suggest changes to improve the final application (CM), as is usual in other approaches, this report will suggest changes to correct the PIM (Fig. 1 (3A)), the transformation rules (Fig. 1 (3C)), and/or the PSM (Fig. 1 (3B)).

4 Defining the Video Game Usability Model

The term usability has several definitions. In this work, we use the ISO/IEC 25010 (SQuaRE) standard [10] as the basis for defining our Video Game Usability Model. In this standard, three different quality models are proposed: the Software Quality Model, the Data Quality Model and the Quality in Use Model.

The goal of our Video Game Usability Model is to extend the Software Quality Model proposed in SQuaRE, specifically the usability characteristic, for specifying, measuring, and evaluating the usability of video games that are produced throughout a model-driven development process from the end-users perspective. The SQuaRE standard states that the usability of a software product can be decomposed into the following sub-characteristics: *Appropriateness Recognisability*, *Learnability*, *Ease of Use*, *Helpfulness*, *Technical Accessibility* and *Attractiveness*. However, these sub-characteristics are too abstract to be measured in a video game development context.

We therefore propose the decomposition of these sub-characteristics into more representative and measurable usability attributes of video games, and the association of each one of these attributes to specific measures, which can be calculated depending on the characteristics of the software artifact to be evaluated.

4.1 Usability Attributes for Video Game Usability

The decomposition of the aforementioned sub-characteristics into usability attributes is presented as follows, and is summarized in the second column of Table 1. These attributes have been defined by considering and adapting ergonomic criteria for user interfaces [5] as well as knowledge from other domains such as Web development [8], and the underlying usability principles from the existing body of literature in the video game domain [12, 14].

Appropriateness Recognisability contains all the attributes of the video game that ease the understanding of the game. This sub-characteristic is decomposed into the following attributes: *Visibility*, which focuses on visual recognisability, and legibility by measuring the ease of perception of the game's graphic information; *Interface Simplicity* and *Control Simplicity*, which evaluate the complexity of the graphical user

interface and the game controls, respectively; and *Consistency*, which focuses on the degree of similitude and coherence between the elements of the video game.

Learnability contains the attributes of the video game that allow players to learn how to play the game. This sub-characteristic is decomposed into the following attributes: *Feedback support*, which focuses on the game capability to provide information about the current state of the game and its players; and *Tutorial Support*, which verifies whether the game offers a tutorial to teach the players how to play it.

Ease of Use contains all the attributes of the video game that facilitate players' control and operation, both inside and outside gameplay. This sub-characteristic is decomposed into the following attributes: *Control Consistency*, which refers to the degree of semantic similitude of the players' actions with regard to the game controls; *Internal Navigational Simplicity*, which refers to how to navigate between the menu options of a single screen; and *External Navigational Simplicity*, which concerns how to navigate between game screens.

Helpfulness contains the attributes of the video game that provide help when the players need it. This sub-characteristic is decomposed into the following attributes: *Hint Support*, which refers to the game's capability to provide useful hints with which to guide the players; and *Goal Support*, which refers to the video game's capability to provide clear goals for the players to pursue.

Technical Accessibility contains all the attributes that allow physically impaired users to play the video game. This sub-characteristic is decomposed into the following attributes: *Subtitle Support*, which refers to the game's capability to provide adequate subtitles for hearing impaired players; and *Magnifier Support*, which concerns the game's capability to provide adequate sized subtitles for visually impaired players.

Attractiveness contains the attributes that make a video game more appealing to the players. This sub-characteristic is decomposed into the following attributes: *Customization*, which refers to how players can alter the game's graphical user interface and controls to fit their preferences; and *Wait Reduction*, which refers to the degree of inactive waiting the players are forced to undergo.

Table 1. Decomposition of the SQuaRE into measurable attributes and generic measures

Sub-characteristics	**Attributes**	**Measures**
Appropriateness Recognisability	Visibility	Percentage of Screen Usage
	Interface Simplicity	Total Number of GUI Elements
	Control Simplicity	Total Number of Control Mappings
	Consistency	Ratio of Similitude Between Screens
Learnability	Feedback	Total Number of GUI Elements Displaying State Changes
		Ratio of GUI Elements Highlighting State Changes
		Ratio of Meaningful Messages
	Tutorial Support	Tutorial Interactivity
		Tutorial Coverage
Ease of Use	Control Consistency	Ratio of Similitude Between Colliding Game Actions
	Internal Navigational Simplicity	Internal Menu Navigation Depth
		Internal Menu Navigation Breadth
	External Navigational	Shortest Path To Gameplay

Sub-characteristics	Attributes	Measures
	Simplicity	Shortest Path To Exit
		Shortest Return Path To Gameplay
Helpfulness	Hint Support	Availability of Hints
		Hint Understandability
	Goal Support	Goal Visibility
		Goal Understandability
Technical Accessibility	Subtitle Support	Availability of Subtitles
		Subtitle Support for Hearing Impaired Players
		Subtitle Style Differentiation
	Magnifier Support	Subtitle Resize Support
Attractiveness	Customization	Control Remapping
		Interface Customization
	Wait Reduction	Inactive Wait
		Skip Capability of Non-Interactive Content

It is worth to mention that we cannot guarantee that our usability model covers all the possible usability attributes for the video game domain. Our model is an attempt to operationalize subjective heuristics, usability guidelines and recommendations into usability attributes that can be quantified by means of measures. We focused on a set of usability attributes identified by the domain experts.

4.2 Generic Measures for Video Game Usability

Once the measurable usability attributes have been identified, generic measures are then associated with these attributes in order to quantify them. The measures are generic in order to ensure that they can be operationalized in different software artifacts (from different abstraction levels) from different model-driven video game development methods. For the sake of simplicity, only one of the proposed measures from the Video Game Usability Model is presented in Table 2. All the generic measures are summarized in the third column of Table 1.

Table 2. An example of measure from the Video Game Usability Model

Measure	Shortest Return Path To Gameplay (SRPTG)
Attribute	Ease of Use / External Navigational Simplicity
Description	Minimum number of screens that players have to navigate in order to restart the game when the game is over
Formula	Minimum number of steps between the game over screen and the gameplay screen
Scale	Integer greater than or equal to 0
Interpretation	A value of 0 signifies that the game has no menu screens, and players can directly restart when the game is over. Higher values indicate that the players have to navigate many screens before restarting the game.

5 Applying the Video Game Usability Model

In order to apply the Video Game Usability Model to a specific model-driven video game development, we follow a usability evaluation strategy. A typical video game development process consists in the following activities: requirements specification, game design, implementation, and playtesting, along with the usability evaluation. The usability evaluation is conducted by applying the following steps:

1. The Establishment of Evaluation Requirements. The *purpose of the evaluation* as well as all the factors that will condition the evaluation of the game are determined in this phase. *Evaluation profiles* are chosen in order to specify which model-driven game development method is employed, which type of video game is developed, what the target technological platform is, and at which target players the game is aimed. Given a specific model-driven game development method, *software artifacts (models)* and *usability attributes* from the Video Game Usability Model are selected to perform early usability evaluations. The measures associated with the selected attributes are operationalized.

2. Early Usability Evaluation. In this phase, each selected video game software artifact (model) is evaluated according to a set of measures. Each measure provides a numeric value within a specific threshold that indicates whether there is or not a usability problem in the video game. A usability report is consequently generated.

3. Usability Evaluation In-Use. Even when early usability evaluation is performed on video game software artifacts (models), the game should also be further evaluated from the end-users (players) perspective in a specific context of use. This usability-centered playtesting is well documented in the game community [9]. Since this paper focuses on early usability evaluation in model-driven development, usability evaluation in-use is not within the scope of this work.

After usability evaluations, game designers should perform changes to the models in order to solve the usability problems. Early usability problems detected in the game design can be corrected in each model of the corresponding development stage (e.g., PIM, PSM) prior to the code generation.

6 Conclusions

This paper presented a usability inspection method that can be used in early stages of model-driven video game development. The method relies on a usability model that has been developed specifically for the video game domain. This model is aligned with the SQuaRE standard and allows the evaluation of the usability of video games developed according to a model-driven development process.

The inherent features of model-driven development provide a suitable context in which to perform usability evaluations since usability problems that may appear in the final application can be detected and corrected at the model level. Model-driven development also allows automating common usability evaluation tasks that have traditionally been performed by hand. The proposed usability inspection method can also be integrated into any model-driven video game development processes by establishing the relationships between the generic measures from the usability model and the modeling primitives of the different software artifacts of the selected development process.

Nevertheless, we are aware that a further comparison with the users' perception on the usability of a video game obtained by applying a model-driven development process is needed. For this reason, we are currently performing an empirical study to compare the predicted usability of two video games measured using a set of measures from the Usability Model with the perceived usability of these video games measured using a modified System Usability Scale (SUS) [6].

Future works include the empirical validation of the proposed measures with the participation of game software developers, and the empirical validation of the usability model's use in a real industrial model-driven game development project.

Acknowledgments. This research work is funded by the MULTIPLE project (MICINN TIN2009-13838), the FPU program (AP2007-03731) from the MEC-Spain.

References

1. Abrahão S., Iborra E., Vanderdonckt, J.: Usability Evaluation of User Interfaces Generated with a Model- Driven Architecture Tool. Maturing Usability: Quality in Software, Interaction and Value, Springer, pp. 3-32 (2007)
2. Abrahão, S., Insfran, E.: Early Usability Evaluation in Model-Driven Architecture Environments. In: 6th IEEE International Conference on Quality Software (QSIC'06), Beijing, China. IEEE Computer Society, pp. 287-294 (2006)
3. Ampatzoglou, A., Stamelos, I.: Software engineering research for computer games: A systematic review. In: Information and Software Technology, Volume 52, Issue 9, pp. 888-901, ISSN 0950-5849, DOI: 10.1016/j.infsof.2010.05.004 (2010)
4. Andre, T.S, Hartson. H.R, Williges. R.C: Determining the effectiveness of the usability problem inspector: a theory-based model and tool for finding usability problems. Human Factors 45(3): 455-82 (2003)
5. Bastien, J. M. and Scapin, D. L.: Ergonomic Criteria for the Evaluation of Human-Computer Interfaces, version 2.1 (1993)
6. Brooke, J.: SUS - A quick and dirty usability scale. In P. W. e. a. Jordan (Ed.), Usability Evaluation in Industry, 189-194. London: Taylor & Francis (1996)
7. Federoff, M.: Heuristics and Usability Guidelines for the Creation and Evaluation of Fun in Video Games. Indiana University Master of Science Thesis (2002).
8. Fernandez, A., Insfran, E., Abrahão, S.: Integrating a Usability Model into a Model-Driven Web Development Process. 10th International Conference on Web Information Systems Engineering (WISE 2009), pp. 497-510, Springer-Verlag (2009)
9. Greenwood-Ericksen, A., Preisz, E., Stafford, S.: Usability Breakthroughs: Four Techniques To Improve Your Game. In: Gamasutra (2010), http://www.gamasutra.com/view/feature/6130/usability_breakthroughs_four_.php.
10. ISO/IEC 25010: Systems and software engineering, Systems and software Quality Requirements and Evaluation (SQuaRE), System and software quality models (2011).
11. Järvinen, A., Heliö, S. Mäyrä, F.: Communication and Community in Digital Entertainment Services. Prestudy Research Report, Hypermedia Laboratory, University of Tampere, Tampere (2002), http://tampub.uta.fi/tup/951-44-5432-4.pdf
12. Microsoft: Best Practices for Indie Games 3.1, http://create.msdn.com/en-US/education/catalog/article/bestpractices_31.
13. Nacke, L.: From Playability to a Hierarchical Game Usability Model. In: FuturePlay at Game Developers Conference Canada, Vancouver, Canada (2009)
14. Nokia: Top Ten Usability Guidelines for Mobile Games. In: Design and User Experience Library v2.0, http://library.forum.nokia.com/topic/Design_and_User_Experience_Library/top10_usability.pdf
15. Pinelle, D., Wong, N., Stach, T.: Heuristic Evaluation for Games: Usability Principles for Video Game Design. In: Proceedings of the Special Interest Group in Computer Human Interaction (SIGCHI'08), Association for Computing Machinery, pp. 1453–1462 (2008)