Florian Mehm, Stefan Göbel, Ralf Steinmetz: *User Support in Digital Educational Game Authoring Tools*. In: Sybille Hambach, Alke Martens, Djamshid Tavangarian, Bodo Urban: Proceedings of the 3rd International eLBa Science Conference, p. 202-211, Fraunhofer Verlag, July 2010. ISBN 978-3-8396-0135-8.

User Support in Digital Educational Game Authoring Tools

Florian Mehm, Stefan Göbel, Ralf Steinmetz

Fachgebiet Multimedia Kommunikation
Technische Universität Darmstadt
Rundeturmstr. 10
64283 Darmstadt
Florian.Mehm@kom.tu-darmstadt.de
Stefan.Goebel@kom.tu-darmstadt.de
Ralf.Steinmetz@kom.tu-darmstadt.de

Abstract: The use of digital games for educational purposes combines the inherent positive characteristics of games (e.g. large acceptance by younger users) with the goal of educating. As digital educational games (DEGs) are a subset of digital games, their creation often requires teams of specialists from diverse disciplines. In order to allow the creation of storytelling-based educational games by non-specialists, we present an authoring approach in which authors work with models of gameplay that abstract from implementation details. The focus of this paper is an analysis of methods for supporting authors in the task of creating a DEG in an authoring tool using this approach. Among the processes are the utilization of the structures defined by such models, validation and completeness checks, structural templates as well as iterative design.

1 INTRODUCTION

Digital educational games have long been available on the consumer market and are receiving more and more attention as a tool to be used in all educational settings. One limiting factor for their introduction into an educational setting is the complexity of game creation. Since DEGs are a subset of digital games, their creation requires similar teams of specialists from various disciplines working together, including programmers, game designers, artists and more. If the costs and personnel requirements associated with the creation of a game are not available in a given setting, the only available option is to use an already created game, which is either fixed or only allows minor customization.

In the development process of DEGs, workflows of game designers, programmers and instructional designers are often realized in different tools; synchronization between these two groups leads to reduced quality and increased development times. Since it leads to a structured authoring process which includes support for handling of the instructional design of the game, the approach presented in this paper is also applicable in this case, where it is desirable to integrate the workflows of different user groups into one shared tool,.

In order to allow non-specialists and especially non-programmers to work on digital educational games, an authoring tool can be envisioned, similar to tools commonly found in the eLearning sector, which shields the user from implementation details such as game engines or file formats, and which allows working on a game on a high level of abstraction. To provide a use case, a teacher might wish to integrate a sequence of images into a DEG. If the game was created from scratch or in a general-purpose game creation tool, several various ways in which an image could be integrated exist, for example as a texture for a 3D object or a 2D (animated) sprite and many more. In an eLearning tool, the semantics of integrating an image would be unambiguous in comparison, as the image usually would only be inserted into the layout of the current page edited.

In this paper, we present a methodology for constructing models of games which fulfill the requirement of allowing working with them on a high level of abstraction, while offering an author the possibility of integrating content and modifications. The presentation of this methodology in section 3 is followed by an analysis of user support techniques in section 4. In section 5, the current state of development of an authoring tool and associated runtime environment based on these concepts is presented.

2 RELATED WORK

Related work for the approach presented herein can be found in the areas of eLearning authoring tools, game creation tools as well as interactive storytelling due to a focus on storytelling-based DEGs.

E-Learning authoring tools are in place at a multitude of educational institutions as well as companies, and can be used to create educational courses which are presented to users for example via the internet as Web-Based Trainings. An overview of tools from this field is given in [HH03].

In the field of tools for the creation of games, commonly used products are often of commercial nature. These tools can be sorted into a spectrum ranging from very complete tools which allow the creation of any kind of game (for example, the Unity game creation suite[1]) to tools linked to one genre or even only one game, which are typically more restrictive. A scientific approach related to the work presented here is the e-Adventure system [TM+09].

Interactive Storytelling systems provide users with the rendition of a story and allow them to alter the course of the presented story. Of importance to the research on authoring tools are the findings of the IRIS Network of Excellence [CD+08], which has the study of authoring systems for interactive storytelling as one of its objectives.

In order to lower the threshold of game creation for non-programmers, visual and natural language programming approaches should be considered. These techniques allow non-programmers to create programs by presenting the program logic in more intuitive visual or textual representations which closely resemble natural language. These approaches can be useful when users of an authoring tool have to configure the execution of parts of the authored games. An example is the Scratch programming language [RM+09].

Related to the description of gameplay models in section 3 is the work presented in [MC09], where the process of game design and programming is approached using techniques from model-driven development. In the area of authoring for storytelling-based DEGs, the method of using a visual editor for high-level structuring of a game's narrative described in [MT+10] as a visual domain-specific language is similar to the Story Editor found in our authoring system presented in section 5.

3 BASIC METHODS

In order to provide user support during the process of authoring a DEG, we present two basic methods, the introduction of gameplay models and iterative development.

As a first method, we present the construction of a model for games. This model has several aspects: 1) Capturing the gameplay of a certain game, describing the objects that are found in this kind of game and their interrelations and 2) Modeling the contextual information of game sequences.

For describing the contextual information of game sequences, the term Narrative Game-Based Learning Object (NGLOB) has been introduced in

[1] http://unity3d.com/

[GM+09]. A NGLOB is one basic sequence in a game which is annotated with information about the narrative context (the suitability to appear during certain phases of the game's narrative), the gaming context (the suitability for certain types of players) as well as the learning context (the skills/knowledge imparted in the sequence and those needed to understand the sequence).

The necessity of modeling the gameplay and the objects used in a game has been motivated in section 1. As an example, a model for point-and-click adventure gameplay would usually feature objects such as background images, characters or hotspot regions. Underlying the model are implementation details, such as choosing the right animations to play when a character carries out a certain task.

Such a model, when introduced into an authoring tool, allows an authoring workflow in which the author can initially choose a model to use, and then configure those aspects where the model allows configuration. While this workflow restricts the author in the possible games that can be created, it introduces a separation of concerns, where the author is not required to implement the actual gameplay, but can instead use a configurable implementation. By adapting this workflow, the user is also presented with a structured design space, defined by the possibilities of the used model as well as the modifications the user can make.

As noted above, the model presented here is based on the possibility of structuring in game into several small sequences. When categorizing game genres according to the amount of narrative, pure gameplay and simulation they commonly include, as carried out by Lindley in [L05], it can be observed that the genres which are governed by storytelling are those that comply with this requirement the most. Other genres, such as that of strategy games, typically do not feature such small sequences, but are rather akin to simulations and games with little narrative, which are defined by a starting state and the rules governing the gameplay. Therefore, our focus lies on the authoring of storytelling-based DEGs which naturally can be decomposed into small sequences.

The second technology that we propose as a basis for author support is rapid prototyping of game-based learning applications. Using this methodology, early, prototypical implementations of a game are created, in order to test and receive feedback, which is used in further iterations of design and development. This leads to a process of iterative design (c.f. [Z03]), where a cycle of development and testing is formed. In this context, it is important to receive useful information from these tests, both in the form of feedback by testers and data that can be created automatically, such as the time players spent in a given sequence or the choices most often made if the game's narrative allows branches.

4 AUTHOR SUPPORT

Based on the technologies described in section 3, several methods for assisting authors in the task of creating an educational game are identified below.

4.1 Structured Authoring

If the workflow of an authoring tool for DEGs is based on choosing and configuring a model as outlined in section 3, this workflow can be utilized in several support strategies. One advantage inherent in this workflow is the structured way in which authors can operate. While authors working with a general-purpose game creation tool would have to assemble the structure themselves by laying the necessary foundations for game mechanics and content integration in the game creation tool, authors working with a model can work on a higher level of abstraction, where only the necessary modifications are exposed. When choosing a model to use and later during the authoring process, it should be made clear to the user how the model will be executed on the target platform and which effects the author's modifications have, e.g. as a textual description or a visual preview. During the authoring process, the user can be supported by making the modifiable parts of the model obvious. Similar to the way the Scratch programming language [RM+09] visualizes how code fragments can be combined, these parts can be shown to the user in an intuitive way. For example, models could be visualized as having slots which can only be filled with specific object types.

Concerning the notion of NGLOBs, the models underlying these objects (story, player and learner models) can assist untrained authors by providing further structure. For example, an author untrained in writing a suspenseful game narrative is assisted by choosing from a set of common story models such as the Hero's Journey [C72] which provide a list of story stages which the author can use to structure the narrative of the game.

4.2 Validation and Completeness Checks

Based on the different aspects of game models outlined above, validation and completeness checks can be introduced to the authoring workflow, which is facilitated by the structured authoring process. As an example, if the user has omitted a necessary modification to a model, the system could warn the user or abort upon saving.

The contextual information about game sequences (concerning narrative, learning and gaming) included in the definition of NGLOBs (see section 3) can also be used for validation and completeness checks. The narrative annotations can be used to determine if all stages of a chosen story model have been addressed by the user and if the paths through the game narrative fulfill certain criteria. The information about the suitability for certain player types can be used to inform the author if the supplied sequences are too much focused on one type of player and do not offer enough alternative gameplay options for other player types. Finally, the information about the learning context of the game sequences can be used to generate an overview of the skills addressed in a game and an evaluation if sensible learning pathways through the game exist on which players are ideally never lacking necessary skills or are not challenged.

4.3 Structural Templates

We refer to templates as pre-created structures provided to authors, typically at the beginning of development of a game or at the beginning of development of a part of a game. One example is a structure where a story model is chosen and one sequence for each step defined in this story model is created initially. Templates could also include settings for the gameplay models, similar to tutorial games offered with many game-creation tools. Conceptually, this notion of template is similar to the configuration of gameplay models described above, which can also be construed as templates for gameplay which are filled with details by the author. However, it has to be observed that these two notions differ in some respects. While the concept of gameplay models includes the abstraction from the actual implementation and semantic information about objects (e.g. in which way an image will be used during execution), the notion of templates here is related more to the structure of a game.

4.4 Iterative Design and Rapid Prototyping

An authoring system for DEGs can assist authors by providing them with means of integrating the results of tests into the authoring process. This leads to a cycle in which development alternates with testing, leading to incremental design and development of the game. Two techniques are relevant in this context, the integration of test results into the authoring process and rapid prototyping. Rapid prototyping as described in section 3 can be used to allow authors to get early impressions of the effects of design choices. Also, if the actual gameplay implementation is not yet done, rapid prototyping can be used to check if the models under consideration are appropriate and useful.

During testing, information about the test sessions can automatically be logged into a database. This information can be used by an author to make informed choices about necessary changes or additions to the game. Information could include the sequences executed during a session of the game, timing information or the evolution of the player or learner model over time. This information can be automatically fed back into the authoring tool and be provided to the user. As an example, the average time users spent inside each sequence could be generated from the durations logged in the user sessions in which the last version of the game was used.

5 CONTRIBUTIONS

Our primary contributions described here are the StoryTec authoring platform, which features a partial implementation of the basic technologies and the author support methods described above, as well as a runtime component capable of executing a game created in the authoring environment, which is presented in conjunction with a prototyping platform.

5.1 StoryTec Authoring Tool

The authoring tool StoryTec is based on previous work in the field of digital storytelling, especially in the context of the INSCAPE project [B07]. Built as a modular framework, the main interface of StoryTec is composed of several views on the game under development and the objects available for integration (see Figure 1). The views include the Story Editor in the lower part of the interface, which is used to define the structure of a game in the form of a hierarchical network of scenes, which are connected by transitions which indicate which scenes can lead to which others during runtime. The configuration of individual scenes and the definition of objects inside a scene are done in the Stage Editor in the upper part of the user interface, which functions similar to the WYSIWYG components commonly found in game editors. The objects that are available to the user are placed in the Objects Browser on the upper right of the user interface. Another component used in the authoring process is a visual editor for configuration of execution logic and conditions. For further details on the authoring tool, see [MG+09].

Included in the current state of the authoring tool is the concept of NGLOBs (see section 3). A basic model of gameplay found in a target player application as described in section 3 is included, however, its use and structure is not yet presented to the user in an explicit way as described above. This lead to users in an early evaluation carried out with the tool to be initially unsure which effect choices during authoring would have on the actual execution of the game (c.f. [MG+09]).

5.2 Runtime and Prototyping Platform

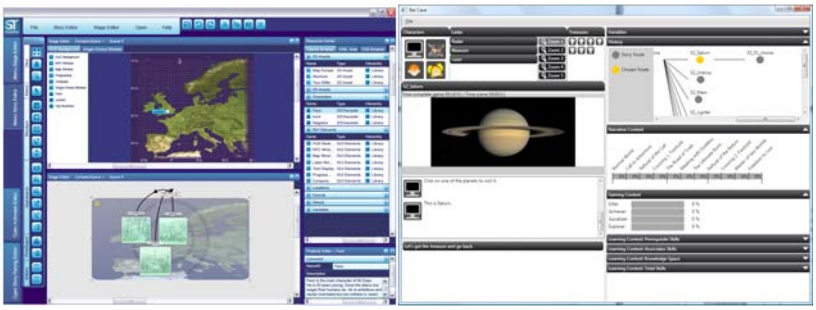

Figure 1: StoryTec Authoring Tool (Left), “Bat Cave” Platform (Right)

The actual execution of games configured in the StoryTec authoring tool is carried out by a specialized parser/interpreter called Story Engine, which is embedded as a component into a player running on the target platform of the game. This engine keeps track of the current state of the game and delegates actual tasks such as displaying different levels or implementing gameplay to the player environment it is embedded into. Due to this architecture, reusability in different player applications is possible, while handling the structures of games exported from the authoring tool is always carried out in a consistent way.

This runtime component is the basis for a rapid prototyping platform for games created in StoryTec, which can be used to design and develop gameplay models. Since the inputs that are handled by the game logic as configured in the authoring process and the outputs in the form of commands to the game implementation are explicitly known, a prototypical implementation of the game can be created. Without any further information, the most basic form of prototype simply logs commands issued by the Story Engine and provides a set of buttons which can trigger the inputs that are handled in the game logic. More advanced prototyping is possible if the gameplay-model-based approach as described in section 3 is used, by implementing these components in only a prototypical fashion.

In the context of the 80Days project [GM+09], a similar approach is currently being used for the creation of a prototypical game implementation to be used for demonstrating the underlying technologies of the project. This prototypical game, termed “Bat Cave”, is shown in Figure 1. While the regular 80Days game is realized in an immersive 3D world, the gameplay in Bat Cave is localized in the left part of the application, using 2D representations of scenes and textual representations of the events taking place in the game. The right part of the application is used for displaying context information, for example the history of user choices or the evolution of the player and learner models.

6 CONCLUSION

In this paper, we presented methods of user support in an authoring tool for storytelling-based DEGs to be used by non-specialists in the field of game creation. The described methods rely on the authoring process being carried out on a model of the game under creation. Furthermore, the model includes annotations relating to several context data associated with game sequences. The current status of an authoring tool implementing these techniques and an associated runtime environment has been presented.

Future work includes deepening the understanding of the models of gameplay used as well as the full realization of the described approach in the authoring tool. Included in the former task is the examination of hierarchical models. For example, the overall model chosen for a given game could be that of a point-and-click adventure game. However, an author might wish to integrate another type of gameplay into one or more sequences found in the game, for example a multiple choice test or a general kind of mini game. It might be desirable to reward an item to the player in case the multiple choice test was completed successfully, thus integrating the result into the gameplay defined by the overall model.

7 ACKNOWLEDGEMENT

The work presented in this paper is partly funded by the Hessian Ministry of Science and Arts (HA-Projekt-Nr. 190/09-20).

References

[B07] Balet, O.: INSCAPE An Authoring Platform for Interactive Storytelling. In: Virtual Storytelling. LNCS 4871. Springer Berlin/Heidelberg, 2007; p. 176-177

[C72] Campbell, J.: The Hero with a Thousand Faces. Princeton University Press, 1972

[CD+08] Cavazza, M.; Donikian, S.; Christie, M.; Spierling, U.; Szilas, N.; Vorderer, P.; Hartmann, T.; Klimmt, C.; André, E.; Champagnat, R.; Petta, P.; Olivier, P.: The IRIS Network of Excellence: Integrating Research in Interactive Storytelling. In: ICIDS 2008. LNCS, vol. 5334. Springer, Heidelberg, 2008; p. 14–19

[Gd+09] Göbel, S.; de Carvalho Rodrigues, A.; Mehm, F.; Steinmetz, R; Narrative Game-Based Learning Objects for Story-based Digital Educational Games. In: Proceedings of the 1st International Open Workshop on Intelligent Personalization and Adaptation in Digital Educational Games, 2009; p. 43-53

[GM+09] Göbel, S.; Mehm, F.; Radke, S.; Steinmetz, R.: 80Days: Adaptive Digital Storytelling for Digital Educational Games. In: Proceedings of the 2nd International Workshop on Story-Telling and Educational Games. STEG'09. CEUR Workshop Proceedings, 2009

[HH03] Horton, W.; Horton, K.: E-learning Tools and Technologies. Wiley, 2003

[L05] Lindley, C.: The Semiotics of Time Structure in Ludic Space As a Foundation for Analysis and Design. In: Game Studies, vol. 5, no. 1, 2005

[MT+10] Marchiori, E.; Torrente, J.; del Blanco, Á.; Moreno-Ger, P.; Fernández-Manjón, B.: A Visual Domain Specific Language for the Creation of Educational Video Games. In: IEEE Learning Technology Newsletter, Vol. 12, No. 1, January 2010 (Special Issue on Game-Based Learning), 2010; p. 36-39

[MG+09] Mehm, F.; Göbel, S.; Radke, S.; Steinmetz, R. Authoring Environment for Story-based Digital Educational Games. In: Proceedings of the 1st International Open Workshop on Intelligent Personalization and Adaptation in Digital Educational Games, 2009; p. 113-124

[MC09] Montero-Reyno, E.; Carsí-Cubel, J.: A Platform-Independent Model for Videogame Gameplay Specification. In: Breaking New Ground: Innovation in Games, Play, Practice and Theory. London: Brunel University, September, 2009

[RM+09] Resnick, M.; Maloney, J.; Monroy-Hernández, A.; Rusk, N.; Eastmond, E.; Brennan, K.; Millner, A.; Rosenbaum, E.; Silver, J.; Silverman, B.; Kafai, Y.: Scratch: Programming for All. In: Communications of the ACM, November 2009

[TM+09] Torrente, J.; Moreno-Ger, P.; Martínez-Ortiz, I.; Fernández-Manjón, B.: Integration and Deployment of Educational Games in e-Learning Environments: The Learning Object Model Meets Educational Gaming. In: Educational Technology & Society, vol. 12, 2009; p. 359–371

[Z03] Zimmerman, E.: Play as Research: The Iterative Design Process. In (Ed. Laurel, B.): Design Research, MIT Press, Cambridge, 2003; p. 176-184