Authoring Environment for Story-based Digital Educational Games

Florian Mehm[1], Stefan Göbel[1], Sabrina Radke[2] and Ralf Steinmetz[1]

[1]Multimedia Communication Lab – KOM, TU Darmstadt
[2]httc Hessisches Telemedia Technologie Kompetenz-Center
Merckstr. 25, 64283 Darmstadt, Germany
{florian.mehm, stefan.goebel, ralf.steinmetz}@kom.tu-darmstadt.de, sabrina.radke@httc.de

Abstract. In this paper, the StoryTec authoring system, developed in the context of the European research project 80Days, is presented as an authoring tool enabling users without programming experience to create digital educational games (DEG) or stories and to integrate content into them. We provide an overview of related software as a motivation for the development of a comprehensive but easily learnable authoring tool for serious games which assists authors in different tasks encountered during the creation process. The architectural basis of the system is described, as well as a structured approach to story-based game authoring using the system, outlining each step of the process. The method and results of an early usability test carried out are presented.

Keywords: Interactive Digital Storytelling, Authoring Tool, Visual Programming, Serious Games, Technology-Enhanced Learning

1 Introduction

The authoring system StoryTec which is currently being developed in the context of the 80Days[1] research project, aims at providing diverse, non-specialist target user groups with a comprehensive yet approachable tool for creating serious games and interactive stories.
In an initial step, different potential user groups have been identified:

- Game designers (e.g. level designers, storyboard authors, concept developers) as the 'creative people' within the multi-step authoring and production process for story-based DEGs.
- Pedagogues and content providers as specialists in learning design, didactics and specific learning topics such as geography in the case of 80Days.

[1] 80Days – Around an Inspiring Virtual Learning World in Eighty Days. EU, FP7, IST, STREP, Challenge 4.1.2 Technology-enhanced Learning. www.eightydays.eu

- Teachers using the StoryTec framework to create small educational games as instruments in their courses.
- Technicians (e.g. game programmers) and content producers (e.g. modellers, graphic designers or audio specialists) who want to use StoryTec as a rapid prototyping environment for development and testing purposes.

Except for the last group, members of these groups are usually characterized by limited programming skills. Therefore, an easy and clear design approach (GUI layout and interaction design) as well as support in the form of templates should be provided. Especially members of the first and fourth group usually have some design background and do use design tools such as products in Adobe's Creative Suite in their daily work – or at least are aware of those tools and have a general idea how to use them. Subsequently, the design should consider typical underlying concepts, layout and GUI elements or interaction metaphors in order to provide a familiar look and feel.
In contrast to these groups, members of the second and third group usually do not have huge experience in those design tools, but are at least familiar with Microsoft Office products such as Word, Excel or PowerPoint.
As can be seen from this comparison, there are different expectations in the different potential user groups and it is necessary to find an appropriate compromise supporting all users as much as possible, which must include alternatives to scripting languages most game authoring tools use.
The focus of this paper lies on describing the authoring component of the overall StoryTec system. For a comprehensive overview and details on story execution see [6].

2 Related Work

The three major areas related to the StoryTec system are middleware software products for game development and e-learning as well as interactive storytelling tools. Furthermore, the system shares properties with visual/natural language approaches to programming. These related areas are presented in this section. A general overview of common practices in multimedia authoring is presented in [1].

E-Learning tools. In this category of tools, many research and commercial products are available which support authors in creating traditional e-learning courses. As an example, the Resource Center [8] features a web-based authoring toolset which can be used to create SCORM-compliant courses. A commercial example which also features the visual programming paradigm is Macromedia Authorware[2].
Game middleware. There exist a multitude of game creation tools with various degrees of complexity. One promising commercial example is the Unity3D[3] game creation software, which features a strong continuity between the authoring phase and

[2] http://www.adobe.com/products/authorware/
[3] http://unity3d.com/

the final game by offering a WYSIWYG view of the game during authoring. Another example which was introduced as a tool for teaching programming is the GameMaker system [17], introduced by Mark Overmars.

Storytelling tools. The focus of authoring tools for interactive storytelling lies on creating interactive experiences which are based on dramaturgic structures. Underlying methods and concepts of StoryTec are based on the theoretical and practical results achieved in the RTD projects art-E-fact (Cyranus authoring environment, see [9]), U-CREATE [7] and INSCAPE[4] [2].

Many current digital storytelling authoring systems make use of the emergent narrative approach, based on virtual agents. These include the Scenejo system [19], which is mainly focused on configuring conversational agents, approaches using rehearsals for demonstrating the envisioned story to virtual characters [14] or authoring during the playing experience [21]. Another project using the emergent narrative approach is the Façade system [15].

Visual/natural language programming. One factor limiting the approachability of the game development tools described above is their reliance on programming languages to customize the flow of events in a game when creating more than very basic games or when straying from simple included templates. Visual and natural language programming approaches can be utilized in this context. A very comprehensive survey of approaches is presented in [11].

Natural language programming allows users to write programs using languages similar to natural languages such as English. A broad overview of approaches is provided in [13]. One example relating specifically to game creation is the “Inform 7” [16] programming language for the creation of interactive fiction stories, in which story objects, scenes and interactions are described in a language resembling English.

Similar to natural language programming, the goal of visual programming languages is to create programs without the use of traditional programming languages. A system related to storytelling is the “Storytelling Alice” system as described in [10], in which a hybrid approach between natural language and visual programming is used to teach programming skills. The appropriateness of graph structures for authoring is supported in [20] in the context of authoring the behavior of virtual characters in the BECool system.

Examining the results of an analysis of related work in the fields of game and learning middleware, it becomes apparent that there exist few tools which integrate workflows for both game development and e-learning content production and integration. During the development of educational games, this results in a disconnection between pedagogues, who lack the skills for the actual game development, and game developers, who are not trained in pedagogy. Authoring tools which can connect the complex interactivity and graphical nature of serious games with support for pedagogical and narrative tasks are not researched in detail yet.

The design goals of the StoryTec system place it in the gap between these two, allowing better cooperation between both groups.

[4] INSCAPE – Interactive Storytelling for Creative People. FP6, IST, IP, www.inscapers.com

3 StoryTec Framework

This section describes the architecture of the StoryTec authoring tool, which acts as a framework for plugin suppliers. Figure 1 shows a broad overview of the components of the system.

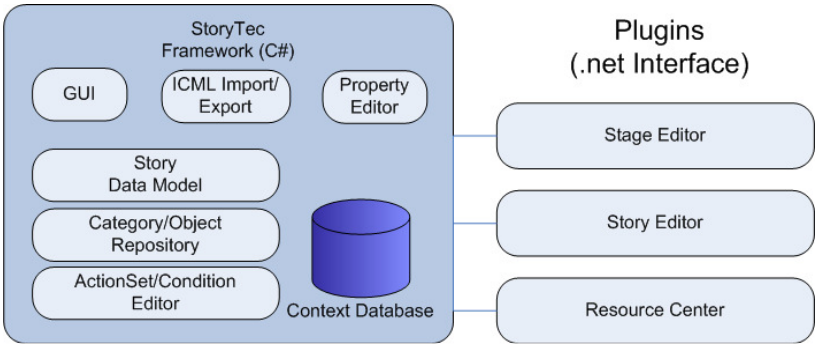


Fig. 1. Architecture of the StoryTec authoring system framework

As can be seen, the core of the system provides a set of central features which can be extended using plugins and which act as a programming framework for the development of plugins. Among the core components are the data model of the application or game the author is working on in the system, an export/import component for serializing the data model into the main export format of the system (ICML – INSCAPE Markup Language – as a story description language, see [6]) as well as a repository of objects available to the author (specific to the target platform, for example 2D or 3D assets, sound files or virtual characters). By providing this core functionality, plugin suppliers are assisted in creating conforming plugins and only need to extend core functionality when it is required (e.g. for providing exporters for platforms which do not utilise the Story Engine [6] which is targeted by the ICML language).

The framework must be augmented by plugins for the remaining core components, which are the Stage and Story Editor and the Resource Center. The Stage Editor component is similar to the WYSIWYG components of many of the authoring tools presented in section 2. Being immediately linked to one or more target output systems, this plugin defines the object categories the user can use in an application for this target platform (e.g. SWF files for a Flash-based player or 3D models for a 3D game engine). The Story Editor plugin is expected to visualize the structure of the story data model, while the Resource Center mainly visualizes the object repository as well as the Context Database, which provides both static and dynamic information such as game assets/props or event logs (see also section 4.5).

While externalizing the Stage Editor as a plugin is essential due to its close relationship to the target platform, the Story Editor and Resource Center were designed as plugins to provide the possibility of addressing new challenges (see the outlook below) while continuing to use older versions.

3.1 Graphical User Interface

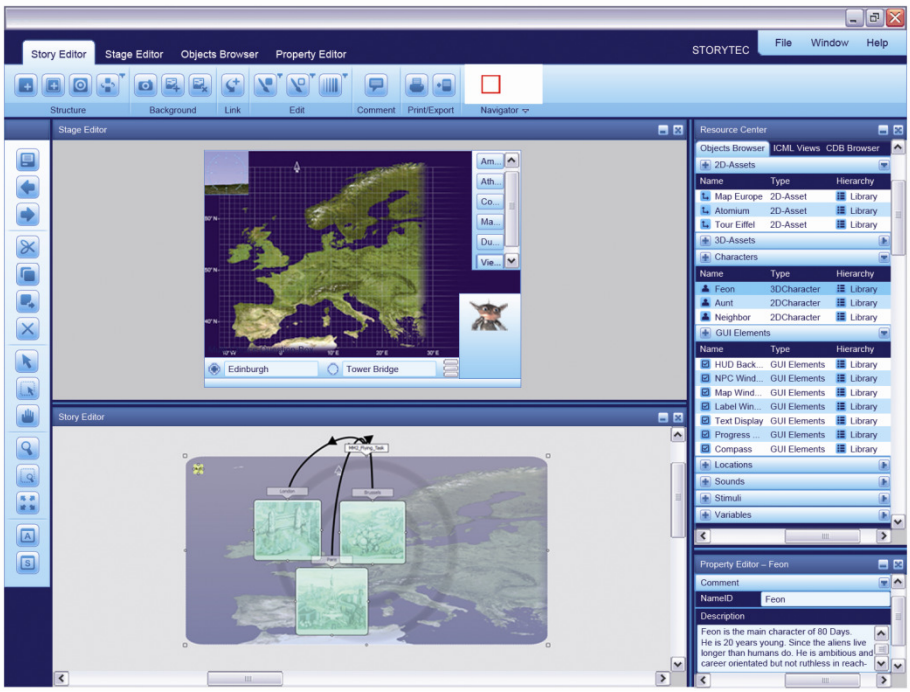


Fig. 2. The user interface of StoryTec (middle-upper part: Stage Editor, right: Resource Center and Property Editor, middle-lower part: Story Editor)

Utilising the WPF[5] framework, the main user interface of the authoring tool as well as styles available to plugins implement a user interface style adapted to the needs of the target user groups as identified and described in section 1. Therefore, an easy and clear design approach should be provided. Hence, the overall aim of the design was to reduce the complexity and presenting the multitude of functions the software provides in a simple and uncomplicated fashion.

As a result, the basic functions are united in the tool menu, which is reached via tabs in the top of the main window. The tool menu is context sensitive, adjusting its functions to the individual editors. This prevents the users from being inundated with information they do not need while working on a specific task. Functions shared by the different editors can be reached quickly via the toolbar on the left side at all times. Furthermore, limiting the colour scheme and making the icons more abstract helps in reaching the goal of reduction of visual complexity. To ensure that the users can adjust the user interface to their needs and the available screen space, flexible window functions are implemented.

[5] Windows Presentation Foundation, http://windowsclient.net/

4 Authoring Process

In this section, we present in general terms (i.e. without reference to a specific Stage Editor plugin) one possible structured process of creating a story-based educational game using StoryTec.

4.1 Creating a Story

After starting the StoryTec application, the user has the choice of loading an existing project, creating a new one from scratch or utilising one of the templates which are provided. A template provides the author with a pre-configured structure which is either based on a story model which can assist authors with little dramaturgic writing skills in creating a coherent and complete story (e.g. by conforming to the Hero's Journey story model) or on a project-specific structure which must be complied with (e.g. the structures a certain player component expects).

4.2 Constructing the Story Structure

If the users opted to create a project from scratch, a possible first step is defining the structure of this project. This can be achieved in the Story Editor component, in which the overall story is partitioned into individual scenes or complex scenes (see [6]). By drawing transitions as arrows between story units, the possible paths through the story are defined. Unconnected scenes should be interpreted as being freely combinable in an adaptive modular storytelling fashion, i.e. the most appropriate scene is selected during runtime, taking into account dramaturgic, learning and gaming aspects (see [5]). While other components like the Stage Editor are found in many of the systems described in section 2, the Story Editor as an abstract view on a story is a component that is an extension compared to many systems.
Scenes can be annotated in detail according to several categories. Authors can specify the expected time a user will remain in a given scene or which function of a story model a scene fulfils. A component visualizing which functions of the model are covered and which are missing in the story assists the user in creating a well-formed story. Skills which can be learned in the scene can be added to the annotation, along with skills the user should possess as prerequisites for the presented learning content.

4.3 Configuring Stages

Having defined the overall structure of a story, the user can continue by configuring the details of each scene. The first step towards this lies in defining which objects are placed in this scene. Similar to related tools as described in section 2, the Stage Editor features a view of the current scene in which the objects featured in the scene are visible in a WYSIWYG fashion. Users can drag objects from the Resource Center and drop them onto the Stage Editor or into the scene visualization in the Story Editor.

Objects will then appear and can be manipulated in both editors, with the Story Editor just displaying an abstract visualization.
Stage Editor plugins are not required to follow an exact WYSIWYG approach. For example, the Stage Editor developed for the 80Days project abstracts from the 3D gameplay by offering a 2D map on which geographical locations can be placed and interactions with them can be defined. This is an advancement from the INSCAPE Stage Editor which featured a very close correlation between the authoring phase and the final result.

4.4 Defining Actions in Scenes

After deciding which objects (such as virtual characters or props) will take part in a given scene, authors can define the logic that governs the flow of events in this scene. In comparable tools, this process is often realized either using a scripting language such as Python or a visual programming approach as in [10]. Since the design of StoryTec is primarily geared towards non-programmers, the latter variant was chosen.
In the ActionSet Editor (see Figure 3), the user can enter the sequence of events that will occur once this scene is encountered in the runtime environment by adding actions, represented by boxes, into a tree structure. Actions are always applied to objects which the author placed in the context of the current scene, therefore, only those objects are available in the ActionSet Editor. Furthermore, authors can trigger the transitions drawn in an earlier step, indicating that the active scene at runtime should change.

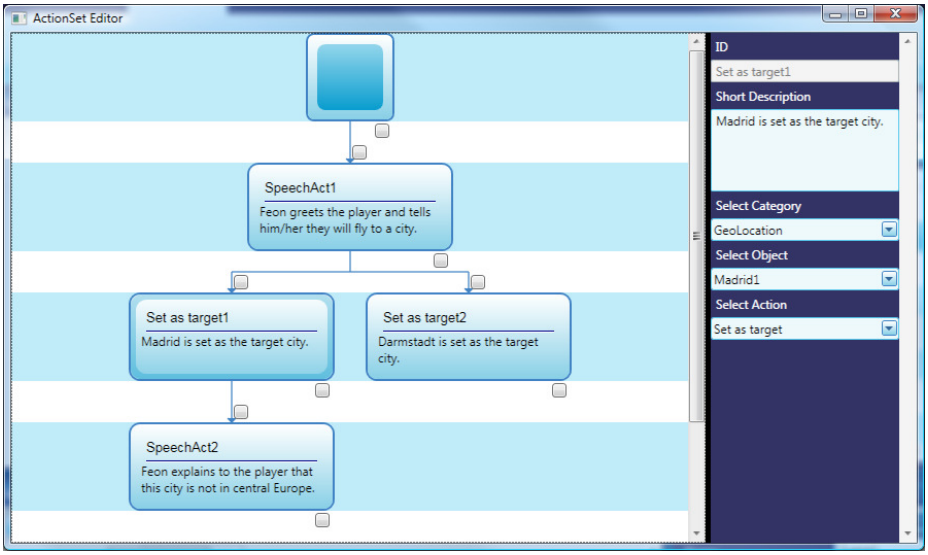


Fig. 3. The ActionSet Editor

In order to be able to react to user input and other events, Stimuli are added to the scenes as a special object type which can also have attached actions.

Branches are created by specifying the conditions for sub-trees. Using the Policy Editor (see [4] for the concept of Policies and Strategies in StoryTec), different game modes (e.g. beginner's/advanced mode or different stress levels) can be created, which can be taken into account when creating conditional branches.
Design challenges for the ActionSet Editor were the paradigms for interacting with the tree structure. For example, if a user deletes one action which is preceded and followed by a condition, the conditions would have to be concatenated in a sensible way. Furthermore, an important question is whether to restrict the ActionSet structure to be a true tree or to allow series-parallel digraphs, which are often used to visualize flowcharts and which are closer to programming languages, since they follow the convention that a program continues after a conditional block regardless of the condition of the block.

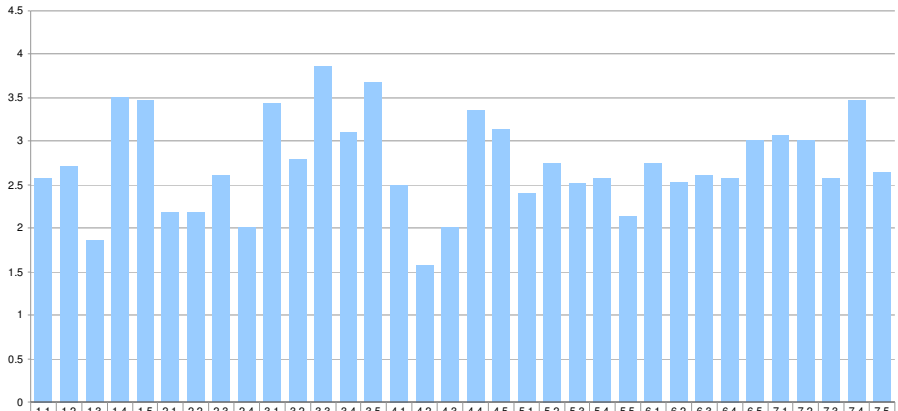
4.5 Testing, Iterative Design

After the first phase of authoring a story-based game has been completed in StoryTec, the game can be exported into the ICML format and be tested in a player application, for example in 80Days using the Nebula2 game engine as player. During testing, various information is saved into the Context Database, including the actual time users spent in a certain scene and the events which took place during a session. This information can then be imported back into the authoring system, where it is consolidated and can be used for example to update the expected time of scenes to be closer to an actual playthrough. This allows an iterative design process (compare with [3]) using which authors can improve their stories over several testing cycles.
The approach of iterative design of games or stories is further supported by offering text-to-speech-functionality to all plugins in the system, in order to test the effect of speech in the project before professional studio recordings are produced.

5 Evaluation

A first evaluation of the currently running StoryTec prototype was carried out with 29 students of which the majority were members of the Computer Science faculty. The participants tested the system in small groups of three. The test consisted of two phases, an usability test, which lasted around 40 minutes, and afterwards a questionnaire. Therefore, the test yielded both qualitative as well as quantitative information.
The first phase of the test was recorded by video and a protocol writer. By using an adapted version of the test method "Thinking Aloud" [18] the participants were asked to choose one person who would operate the system, with the other two instructing that person. By working through a task list, the participants built a small story in StoryTec, which could be played afterwards. By choosing this setup, the participants were encouraged to discuss possible ways of complying with the instructions and the monitoring staff could be made more aware of possible usability problems encountered by the subjects.

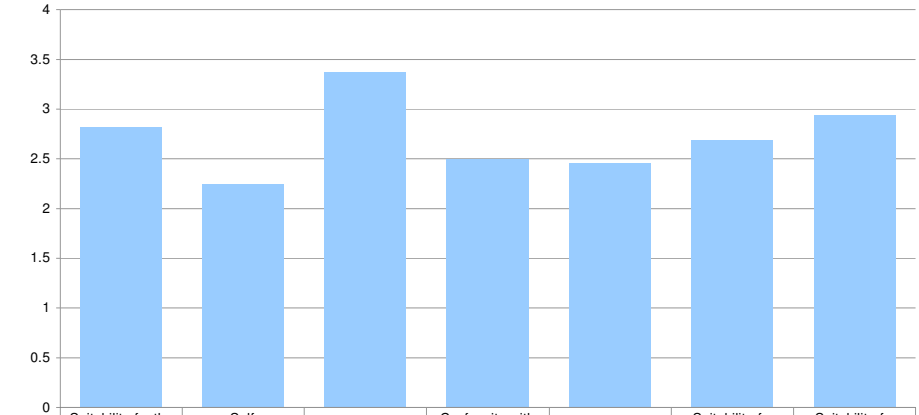
Overall, the interface, the functions and mainly the idea of StoryTec got a very positive rating. StoryTec was received very well by most participants, who described it as an innovative, user-friendly authoring tool. All of them were motivated to fulfil the task until the end. Nevertheless the test aided in discovering a range of small usability problems and technical bugs, which will be fixed in the next step. The result of the qualitative evaluation with the most impact was the lack of understanding the Stage Editor.
Furthermore, the aim of reduction of complexity of StoryTec by using a reduced design (e.g. fewer colours, abstract icons) could not be reached. On the contrary, the first test phase showed that this part of the design lead to problems in understanding the functions and the interface of the tool.
The quantitative test consisted of a questionnaire with 34 items which were presented with a six point scale from very bad to very good. The questionnaire included seven fields of software ergonomics (1. Suitability for the task, 2. Self-descriptiveness, 3. Controllability, 4. Conformity with user expectations, 5. Error tolerance, 6. Suitability for individualization and 7. Suitability for learning) by using the usability standard EN ISO 9241-10.



	1.1	1.2	1.3	1.4	1.5	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5	4.1	4.2	4.3	4.4	4.5	5.1	5.2	5.3	5.4	5.5	6.1	6.2	6.3	6.4	6.5	7.1	7.2	7.3	7.4	7.5
Average	2.57	2.71	1.86	3.5	3.46	2.18	2.18	2.61	2	3.43	2.79	3.86	3.11	3.68	2.5	1.57	2	3.36	3.14	2.39	2.75	2.52	2.57	2.14	2.75	2.54	2.61	2.57	3	3.07	3	2.57	3.46	2.64
SD	1.32	1.46	0.97	1.04	1.32	1.44	1.44	1.37	1.12	1.4	1.29	1.24	1.17	1.12	1.45	1.43	1.27	1.47	1.33	1.45	1	1.12	1.29	1.24	1.35	1.35	1.5	1.29	1.39	1.63	1.41	1.53	1.32	1.66

Fig. 4. Average results of the questionnaire

Figure 4 shows the average answers and the standard deviations of all items. Exemplarily, with an average of 3.86 (SD 1.24), the item 3.3 (addressing the controllability of the system) received the best rating. This implies that StoryTec quietly allows an easy change between menus and masks. The worst rating with an average of 1.57 was given to item 4.2 (addressing the conformity with user expectations of StoryTec). This implies that users were not sure if input they made was correctly executed or not by the system.



	Suitability for the task	Self-descriptiveness	Controllability	Conformity with user expectations	Error tolerance	Suitability for individualization	Suitability for learning
Average	2.821428571	2.241071429	3.371428571	2.5	2.457142857	2.692857143	2.95
SD	0.820730092	0.961003871	0.893953824	0.908397042	0.899147156	1.086594603	1.2479613

Fig. 5. The results of the questionnaire grouped by the 7 fields of software ergonomics as defined in EN ISO 9241-10.

Figure 5 shows the results of grouping the items into the seven fields of the usability standard EN ISO 9241-10. As can be seen, no direct conclusions can be drawn from the results concerning the software ergonomics of StoryTec.
While the evaluation of the questionnaires showed little evidence for certain strong or weak points in the design approach of StoryTec, the protocols of the first phase revealed many important details about the usability of the tool, which are currently being incorporated in an updated design and new paradigms.

6 Conclusion

The StoryTec authoring system is presented as a comprehensive authoring tool which addresses non-programmers and supports them in creating a story-based serious game with a coherent dramaturgic structure. The current version can be used to create projects which are run on the 80Days technical platform using the Nebula2 game engine as player.
Future versions will include several improvements. One area currently being worked on is the development of additional Stage Editors to support more target platforms. Among the possible platforms are web-based players using Adobe Flash, mobile appliances such as Nintendo DS, iPhone or Blackberry as well as 3D players using game engines or frameworks such as Unity3D or Microsoft XNA. The integration of more resources such as learning objects using SCORM is planned, too.
Furthermore, more research will be carried out in several areas. Among them are more advanced visual programming paradigms, especially in the ActionSet Editor, as well as supporting concurrency of Actions. In this context, the Condition Editor for configuring branches will be extended. Similarly, the default Story Editor will be

updated to better visualize important information during different stages of the authoring process, for example by visualizing the structure of the story in reference to the chosen story model or by hiding currently unimportant parts of the story. The visualization of skill structures (see [12]) underlying the learning aspects/goals of the story are planned as well.
In order to optimize the usability of the tool for the target audience, the results of the first usability test are currently being integrated, and new usability tests in later stages of development will be carried out.

Acknowledgements
The research and development introduced in this work is funded by the European Commission under the seventh framework programme in the ICT research priority, contract number 215918 (80Days, www.eightydays.eu).

References

1. Bulterman, D., Hardman, L.: Structured multimedia authoring. ACM Transactions on Multimedia Computing, Communications, and Applications 1(1), 89-109 (2005)
2. Dade-Robertson, M.: Visual Scenario Representation in the Context of a Tool for Interactive Storytelling. In: Cavazza, M., Donikian, S. (eds.) Virtual Storytelling 2007. LNCS, vol. 4871, pp. 3-12. Springer, Berlin / Heidelberg (2007)
3. Fullerton, T., Chen, J., Santiago, K., Nelson, E., Diamante, V., Meyers, A., Song, G., DeWeese, J.: That cloud game: dreaming (and doing) innovative game design. In : Proceedings of the 2006 ACM SIGGRAPH symposium on Videogames, pp. 51-59. ACM, New York, NY, USA (2006)
4. Göbel, S., Malkewitz, R., Becker, F.: Story Pacing in Interactive Storytelling. In Pan, Z., Aylett, R., Diener, H., Jin, X., Göbel, S., Li, L. (eds.) Technologies for E-Learning and Digital Entertainment. LNCS, vol. 3942, pp. 419-428. Springer, Berlin / Heidelberg (2006)
5. Göbel, S., Mehm, F., Radke, S.: 80Days: Adaptive Digital Storytelling for Digital Educational Games. Accepted to Second Workshop on Story-Telling and Educational Games, STEG (2009)
6. Göbel, S., Salvatore, L., Konrad, R., Mehm, F.: StoryTec: A Digital Storytelling Platform for the Authoring and Experiencing of Interactive and Non-linear Stories. In Spierling, U., Cavazza, M., Peinado, F., Aylett, R., Swartjes, I., Kudenko, D., Young, R., Tychsen, A., Pizzi, D., El-Nasr, M. (eds.) Interactive Storytelling 2008. LNCS, vol. 5334, pp. 325-328. Springer, Berlin / Heidelberg (2008)
7. Göbel, S., Salvatore, L., Konrad, R., Sauer, S., Osswald, K.: U-CREATE: Authoring tool for the creation of interactive storytelling based edutainment applications. In Cappellini, V., Hemsley, J. (eds.) EVA 2007 Florence Proceedings. Pitagora Editrice Bologna, Bologna (2007)
8. Hoermann, S., Hildebrandt, T., Rensing, C., Steinmetz, R.: ResourceCenter - A Digital Learning Object Repository with an Integrated Authoring Tool Set. In Kommers, P., Richards, G. (eds.) Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications 2005. pp. 3453-3460. AACE, Chesapeake, VA (2005)
9. Iurgel, I.: An authoring framework for interactive narrative with virtual characters. PHD Thesis, Technische Universität Darmstadt, Darmstadt (2008)

10. Kelleher, C., Pausch, R.: Lessons Learned from Designing a Programming System to Support Middle School Girls Creating Animated Stories. In : IEEE Symposium on Visual Languages and Human-Centric Computing, 2006, pp. 165-172. IEEE Computer Society, Washington, DC., USA (2006)
11. Kelleher, C., Pausch, R.: Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. ACM Computing Surveys 37(2), 83-137 (2005)
12. Kickmeier-Rust, M., Göbel, S., Albert, D.: 80Days: Melding Adaptive Educational Technology and Adaptive and Interactive Storytelling in Digital Educational Games. In Klamma, R., Sharda, N., Fernández-Manjón, B., Kosch, H., Spaniol, M. (eds.) Proceedings of the First International Workshop on Story-Telling and Educational Games. CEUR Workshop Proceedings (2008)
13. Knöll, R., Mezini, M.: Pegasus: first steps toward a naturalistic programming language. In: Dynamic Languages Symposium, Companion to the 21st ACM SIGPLAN symposium on Object-oriented programming systems, languages, and applications, pp. 542-559. ACM, New York, NY, USA (2006)
14. Kriegel, M., Aylett, R.: Emergent Narrative as a Novel Framework for Massively Collaborative Authoring. In Gratch, J., Prendinger, H., Ishizuka, M., Marsella, S., Pelachaud, C., Kopp, S., Kipp, M., André, E., Rehm, M., Aylett, R. (eds.) Intelligent Virtual Agents. LNCS, vol. 5208, pp. 73-80. Springer, Berlin / Heidelberg (2008)
15. Mateas, M., Stern, A.: Structuring content in the Facade interactive drama architecture. In Young, M., Laird, J. (eds.) Proceedings of the Conference on Artificial Intelligence and Interactive Digital Entertainment. AAAI, Menlo Park, CA. (2005)
16. Nelson, G.: Natural Language, Semantic Analysis and Interactive Fiction. Whitepaper (2005)
17. Overmars, M.: Teaching computer science through game design. Computer 37(4), 81-83 (2004)
18. Rosenbaum, S., Rohn, J., Humburg, J.: A toolkit for strategic usability: results from workshops, panels, and surveys. In: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 337-344. ACM, New York, NY, USA (2000)
19. Spierling, U., Weiß, S., Müller, W.: Towards Accessible Authoring Tools for Interactive Storytelling. In Spierling, U., Tychsen, A., Peinado, F., Stern, A., Riedl, M., Hitchens, M., Magerko, B., Manninen, T., Vallius, L., Kujanpää, T. (eds.) Technologies for Interactive Digital Storytelling and Entertainment. LNCS, vol. 4326, pp. 169-180. Springer, Berlin / Heidelberg (2006)
20. Szilas, N.: BEcool: Towards an Author Friendly Behaviour Engine. In Cavazza, M., Donikian, S. (eds.) International Conference on Virtual Storytelling. LNCS, vol. 4871, pp. 102-113. Springer, Berlin / Heidelberg (2007)
21. Thue, D., Bulitko, V., Spetch, M.: Making Stories Player-Specific: Delayed Authoring in Interactive Storytelling. In Spierling, U., Cavazza, M., Peinado, F., Aylett, R., Swartjes, I., Kudenko, D., Young, R., Tychsen, A., Pizzi, D., El-Nasr, M. (eds.) Interactive Storytelling. LNCS, vol. 5334, pp. 230-241. Springer, Berlin / Heidelberg (2008)