Evaluation of Adaptive Serious Games using Playtraces and Aggregated Play Data

Christian Reuter, Florian Mehm, Stefan Göbel, Ralf Steinmetz
TU Darmstadt, Darmstadt, Germany
christian.reuter@kom.tu-darmstadt.de
florian.mehm@kom.tu-darmstadt.de
stefan.goebel@kom.tu-darmstadt.de
ralf.steinmetz@kom.tu-darmstadt.de

Abstract: Adaptive Serious Games often feature complex algorithms and models, which influence the player's progression through the game. These models include properties like pre-existing knowledge or preferred playstyle and are matched with a pool of appropriately annotated parts of the game, such as assignments or scenes, during runtime. While being transparent for players, these models must be visualized for testing and evaluation purposes.

In order to allow authors the retrospective interpretation of playtraces generated by a gaming session, we developed a replay component for adaptive serious games created with the authoring tool "StoryTec". This method removes the need for continuous observation of individual players while retaining the same level of detail and being much more understandable compared to log files, especially for the non-programming audience addressed by StoryTec. In addition to showing the player's view, the state of the internal models and the progression through the story structure are also visualized. Sharing the same models and data structures as the authoring tool and making their runtime behaviour visible to the author, the replay component is therefore able to offer additional benefits compared to more generic methods like screen capturing or key recording tools.

A complementary tool which is able to aggregate a large number of playtraces into one comprehensive spreadsheet for statistical analysis was also implemented. This allows authors to gain an overview over a great number of players in a shorter time compared to investigating them individually. In order to reduce the complexity of the result, the table contains aggregated information like the total time the players spent in each scene or the final value of variables at the end of their sessions. If authors detect an anomaly, they can then access more detailed information by loading the original traces into the replay component, which uses the same data format.

Together these two components support the evaluation of adaptive serious games by means of user studies with the intended target audience, for example pupils. By combining them with our testbed for rapid prototyping named "StoryPlay", we were able to provide a set of tools covering a broad range of evaluation tasks based on the same underlying models and data formats. Using these tools, it is possible to gain insights on how the adaption algorithms behave over a large number of players, e.g. which paths were taken by how many players or whether the time to solve a task as estimated by the author was matched.

Keywords/Key Phrases: serious games, evaluation, adaptation, playtrace, testbed

1. Introduction

While the idea of Serious Games, i.e. games which serve an additional purpose besides entertainment like education or health, becomes more and more popular, there are still critical voices who question the effects of these kinds of games. Therefore it is necessary to provide hard evidence in this regard in order to fully establish this idea, especially when those games are meant to provide game-based learning and training.

This is usually done by means of evaluation studies, where the game is played by a large number of players and their results and / or gameplay experiences are recorded. Besides assessing the outcome of the games, user studies also allow the game's creators to identify technical as well as design problems and measure the acceptance by the intended target audience. There are already many generic tools and methods available that support user studies of software in general as well as games and Serious Games in particular.

Adaptive Serious Games are games that contain adaption mechanism that allow the game to alter itself depending on the player using it. This can happen by simply personalizing the presentation based on the user's preferences or by taking his pre-existing knowledge into account and restructuring the order and overall number of assignments / game scenes accordingly. When evaluating adaptive games, it is important to not only record the player's view, but also the state of the underlying adaption models and algorithms, giving insight into how these behave over a wide range of users. An obvious benefit of this approach would be a case, where an adaptive game with multiple paths through its story is evaluated, showing that some paths are never chosen by the adaption engine. In order to assess whether this is an error and how to fix it, a close observation of the internal states is necessary.

The current state-of-the-art regarding evaluations can be separated into two groups of methods. Generic methods and tools like observation, screen capturing or questionnaires only record the "outsiders view" and therefore work with arbitrary games. In contrast, specific approaches must be tailored towards a concrete game, but allow the recording of internal information. To fully evaluate adaptive games, it is therefore necessary to employ specific methods, which are closely linked to their internal models.

A drawback of specific methods however is that they offer limited reusability and often must be adjusted for each game they are applied to, creating a trade-off between the amounts of information an evaluation tool is able to provide and the costs associated with its use. In order to address this problem we therefore developed two tools based on the adaption models of our authoring tool "StoryTec" (Göbel et al. 2010). Since the games created with it are based on the same adaptation models, it is possible to evaluate all of them using the same set of tools. The tools also aim at reducing the effort for evaluating complex games with many players and are intended to be easily used by single non-programming authors.

In this paper we first discuss the current state-of-the-art in regards to evaluation methods and tools for adaptive Serious Games. We then give a short overview over the authoring tool "StoryTec" and the associated "StoryPlay" Testbed, which build the foundation for this work. After that we describe our evaluation framework, consisting of a replay component and an aggregation tool for playtraces. We also show how our tool is used in the ongoing evaluation of a learning game for mathematics, focussing on explorative results like the overall playtime or anomalies in player behaviour.

2. Related Work

A good overview of software evaluation in general can be found in (Hilbert & Redmiles 2000). They argue that user interfaces automatically generate events that can be recorded and describe different types of evaluation goals and event data. Based on this the authors compared tools and techniques, which are able to analyse these recordings and can extract higher level information from them. Examples include the synchronization with other data sources such as video recordings, transformation (e.g. filtering), summarization (to decrease the amount of data) and the detection / comparison of subsequences of events.

Another overview aimed specifically at games was done by (Nacke et al. 2009), differentiating between playability, which focusses on the game itself, and player experience, which concerns the player's interaction with the game. They argue that good playability is necessary to conduct studies on player experience. The work then lists a number of methods for both types of evaluations. While playability is measured against heuristics, player experience can be assessed by a number of different objective (like biofeedback, gameplay data) and subjective methods (questionnaires). It is concluded that a combination of several methods yields more information and is therefore advisable. (Nacke et al. 2010) follows up on the topic of player experience in the regards to Serious Games. They also list several evaluation methods, grouped in individual (e.g. psychophysiological, gameplay data, player modelling and questionnaires) and context oriented (playability heuristics, also questionnaires) ones that take the environment in which the game is played into account. (Bruder et al. 2004) describes a certificate for computer-based learning environments, which also takes the learning effects and outcomes into account – a criterion for Serious Games in particular.

Individual approaches that work with games in general include work by (Ketkar & Youngblood 2010), where the movement of players in a 3D world was recorded. After that graph based algorithms were

used in order to build player profiles. (Liu et al. 2011) used heatmaps to visualize players' progress in games with discrete states, needing only the states and transitions between them as an input information. They also proposed the mapping of continuous spaces to state features, making their approach viable for games without discrete states while also defining higher level information. This mapping however must be done manually and the results of the visualization are highly dependent on the mapping. (Kim et al. 2008) developed a testing framework which can be used to interpret user triggered events, which are logged by a game. It is able to visualize anomalies on different levels of granularity. When linked to a video recording, they were able to find the reason behind sudden difficulty spikes and changed their game accordingly. They noted that it is important to decide which events to record, that the sequence of events is very important and that only the number of occurrences is often insufficient.

One example where data analysis was conducted for a commercial game can be found in (Medler et al. 2011). They used a graph based visualization of events to balance two heterogeneous teams in a multiplayer setting. Their server-based system was directly integrated into the development process. Practical insights for the development of such tools were noted and clustered in the categories production (for example to build the tool parallel to the game), functionality (enabling faster analysis by aggregation) and game team integration (involving the game team in the tool development process). Other work by (Hullett et al. 2012) gathered game data from real players spanning 3 years after release of a game via the internet, requiring no additional collection setup. Their analysis produced insights for future games, such as popular or unused game modes and elements – to pick their focus for future releases to decrease development costs.

Another visualization tool for game data is "StoryPlay", formerly "Bat Cave" (Mehm et al. 2010). This tool is tailored for adaptive Games created with the authoring tool "StoryTec" and that visualizes the internal game state. Since the tool is closely linked to this work, it will be discussed in the next chapter.

3. Background

The evaluation framework we describe in this paper is based on the internal models of the StoryTec-authoring environment on the one hand and on the visualizations of the StoryPlay-testbed on the other hand.

3.1 Adaptive Games in StoryTec

Games – or stories – build in StoryTec offer adaptation along three dimensions: the learner, player and story model (Göbel et al. 2010). The learner model takes the pre-existing knowledge and dependencies between individual skills into account. Its main task is to make sure that the players do learn all skills taught by the game while guaranteeing that the learner has the right perquisites to understand this knowledge. To represent the dependencies between individual skills, the learning context is modelled as a graph based on the "Competence based Knowledge Space" (Korossy 1999).

The player model is based on (Bartle 1996) and maps the players behaviour to four playstyles, "killer", "achiever", "socializer" and "explorer". This allows different representations of same content, fitting the players preferred style of play. Each of these styles is modelled by a number in the interval [0, 1], based on how well the style fits the player.

Lastly the story model uses a modified version of the Hero's Journey (Göbel et al. 2009) to balance between adaptation and an interesting story as defined by the author. The fixed order of event types (like "call to adventure" or "return") from the Hero's Journey is preserved while allowing the events themselves to play out differently.

During game creation, the author is able to annotate every scene along these three dimensions, for example describing which learning content they provide and thereby making them "Narrative Game-based Learning Objects". When the game is played, these models are constantly updated based on the players progress and behaviour by the Story Engine. Whenever a free transition to another scene is triggered, the engine calculates a score for every potential successor based on their annotation and the current state of the models as well as individual weights for each dimension. The scene which fits the player best is then selected.

3.2 Testbed StoryPlay

When prototyping adaptive games created with StoryTec, it was necessary to visualize the state of these internal models. This made it possible to understand decisions of the Story Engine and, if the results seemed undesirable, change the game or the adaption algorithms accordingly.

Since the models may seem very complex, especially for single untrained authors, we developed a rapid prototyping environment which is able to visualize the state of the models and the decisions of the Story Engine in an understandable way (Mehm et al. 2010). The testbed works directly on the story files created by StoryTec and is linked to the same models, reducing effort for rapid prototyping and eliminating the risk of mismatches or loss of information between the two tools.

4. Evaluation Framework

Most evaluation methods described in the related work require internal play data from the game, which should include the state of the internal models in adaptive Serious Games. Since this data is already visualized by the testbed, our first approach used StoryPlay for small user studies. When working with practitioners it became clear however, that a rapid prototyping session by an author has different requirements than a general evaluation tool:

- Authors usually work alone or in small groups, while evaluations ideally feature large groups of players.
- Authors need lots of information regarding the internal models, while players participating in an evaluation get distracted by them.
- Authors want to see the state of the models in real-time, but during evaluations the models will be analysed retrospectively.
- For rapid prototyping the amount of information is limited to that generated by one gameplay session at a time, while in evaluations an overview over a number of sessions is needed.
- For rapid prototyping information has to be presented during play, while in an evaluation setting the information can be transformed before it is presented.

In order to address these requirements, we developed two additional tools that complement the authoring tool and testbed. The first one is a replay component, which was directly integrated into the testbed to give it the ability to review individual game sessions after they were played. This extension is accompanied by an aggregation tool, which combines the data from a large number of individual sessions into one spreadsheet for further analysis.

Both new tools work with playtraces that are written to log files during a normal play session, where the internal models are hidden from the player. They use the same format / timestamp and are linked to the original story files as well as the models used by StoryTec, so there is no mismatch between these new components either. Therefore it is possible to analyse the aggregated view of the spreadsheet and then investigate anomalies in detail with the replay component without losing information.

Based on the research questions asked in our previous evaluations of adaptive Serious Games, we decided that the playtraces had to contain at least the following information:

- Change of the internal adaptation models along each dimension.
- Time taken per scene (for example explanation, tasks, help-screen).
- If a scene was visited and how often.
- User input (multiple choice, text, minigames and mouse movement).

We also decided to log every stimulus, which notes the execution of an event in the game. This will allow us to always reconstruct the whole playthrough in combination with the original story files. This way there will never be a loss of information, even if a feature was added without logging its information explicitly.

4.1 Replay Component

The replay component allows the testbed to replay game sessions recorded in playtraces retrospectively. It offers the same level of detail like a direct observation of the player, including mouse movements (Figure 1, left). In contrast to the original play session however, the state of the internal models is also visualized (Figure 1, right).

The advantages compared to a direct observation are that the replays can be watched independently from the original game session and more than once. The internal game information and the evaluation setup is invisible for the player and does not require any additional effort besides the collection of the playtraces, which could even happen automatically while the game is played on a website. Compared to generic approaches like capturing a video feed, the playtraces can be much smaller by referencing simple events and having the replay component calculate the same result as the original game – which is only possible when tailoring the component to a specific game or game type.

Additional features of the tool include remote observation during live play and a check whether the story used during generation of a particular playtrace matches the version used during the original play, so no misleading results based on minor variations can be generated.

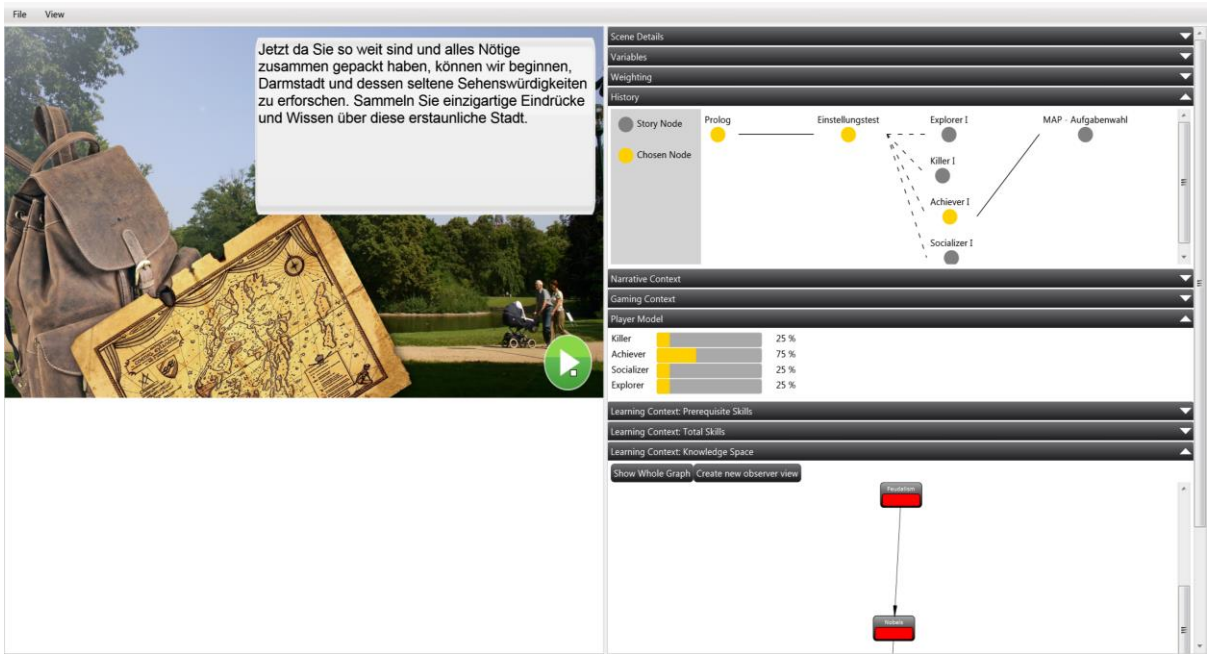

Figure 1: Screenshot of the replay component, showing the players view (including his mouse cursor symbolized by a white box) on the left and the internal state of some of the models (scene history, player model and parts of the knowledge space) on the right.

4.2 Aggregation Tool

The aggregation tool in contrast is designed to provide an overview over a large group of players. Its task is to parse a number of playtraces created by a game and compile this data into one spreadsheet, which can be further analysed using common office tools. Calculating an average score for example does not require opening each playtrace separately, saving time and enabling large scale evaluations for single authors.

By specialising the tool in reading our playtrace format, we were able to exploit the semantic of certain information, for example for aggregation. It is also able to distinguish between temporary and final values of the internal models and variables, although the event might technically be the same. The same differentiation can be done for answers a player has given. Important “higher level” questions that can be investigated using the tool include how the internal models evolve and if they converge over time. It can detect how often scenes were reached, giving the author feedback whether the adaptation produced different results for his players. It is also possible to detect unnecessary scenes, which are seldom reached, giving him a clue on where to focus when refining the game. Based on

completion time, he can also estimate the difficulty of the game and gather information on the expected time for certain types of tasks or minigames, providing data for the creation of a game designed around a certain play time (e.g. during a school lesson).

However certain information cannot be extracted automatically, because there are universal concepts which might be used differently in structurally similar games. One example showed us that an author used generic buttons to model multiple answers to task and to skip the task altogether. Since these buttons can be used in many different ways, we could not differentiate between solved and skipped tasks in a way that could be used over every game created with our authoring environment. When using a predefined template with a fixed semantic for such tasks however, the tool is even able to detect whether a task was solved correctly or not.

In order to support very large games, we implemented a function which is able to filter certain events based on annotations. The aggregation tool also displays a warning, if multiple variations of the same story were used and splits the results accordingly to reflect the differences. We also added anonymous user IDs, which allowed the matching of play sessions to external data source like accompanying questionnaires. These IDs are checked for uniqueness to prevent duplicate entries while complying with privacy regulations by being arbitrary identifiers.

5. Use Case: Evaluation of learning game for mathematics

The first applications where our evaluation framework was used was a game called “Der Wechsel” (“The Switchover”), created for teaching about mathematic functions in school. The game combines the everyday use of math with elements of a crime story and optional minigames like puzzles or hidden object games. It was evaluated with multiple school classes of about 30 pupils each (age 14-15, 55% male), where the pupils had roughly one hour to solve the game on their own. Afterwards the playtraces were collected and analysed using the aggregation tool. We will focus on explorative results in this paper, noting some interesting findings in regards to the overall gameplay, which we were able to quickly isolate by combining the aggregated play data with the replay tool.

Since the game’s design did not use the adaptive features of the authoring framework, the adaptation models stayed constant over the course of the game. We could however verify that the evaluation workflow worked smoothly while including non-expert authors and that even the analysis of non-adaptive variables could produce helpful information, some of which will be discussed in the following paragraphs.

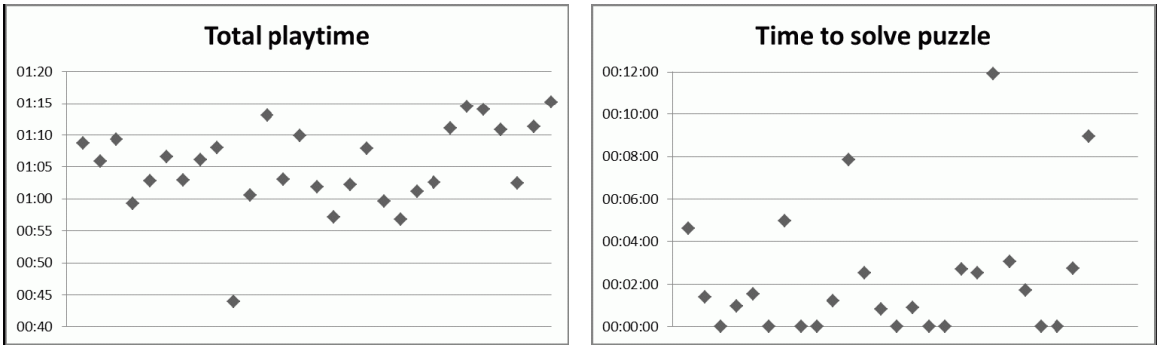

Figure 2: Graphs displaying the total playtime of each pupil (left) and the time it took them to solve a puzzle minigame (right).

While the game was supposed to last one hour, analysis of the total playtime showed that most pupils needed about ten additional minutes to finish the game (Figure 2, left). However on outlier was obvious, who took less than 45 minutes. Further analysis with the replay tool then revealed that the pupil skipped all optional minigames and took lucky guesses at a lot of tasks instead of solving them, indicated by quick and often wrong answers when others paused to calculate their answers.

Analysis of the time the players took to solve a puzzle minigame also yielded valuable feedback (Figure 2, right). Some players took up to twelve minutes to solve the minigame alone, although the game’s authors estimated that completing the puzzle would only take up to one minute (it only

consisted of seven parts). Again the replay tool helped to answer this question by revealing that those players had supposedly solved the puzzle much earlier (all parts were in the right order), but were held back by a usability problem: The game would only acknowledge that the puzzle was solved when they aligned the combined parts with the upper left corner of the screen, but they centred them in the middle. In contrast, almost half of the players skipped the optional puzzle altogether in order to focus on the math questions, which is an interesting information when designing similar games in the future.

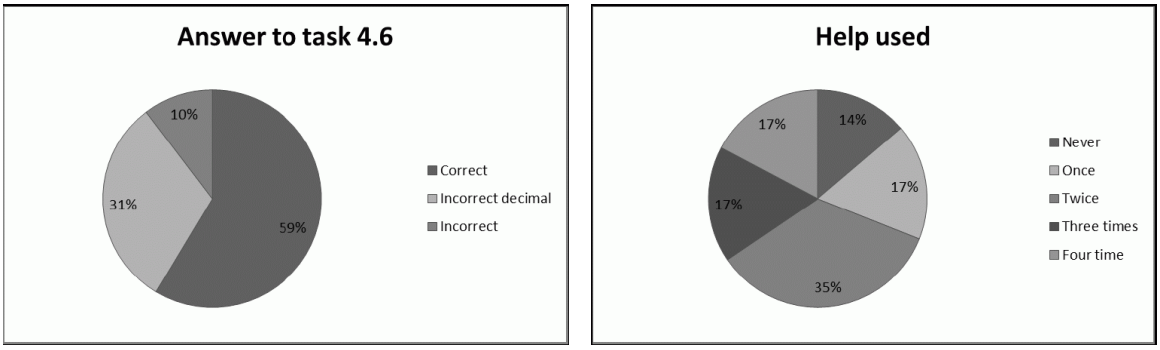

Figure 3: How well a specific task was answered (left) and the number of times the help was used (right).

Other important information is how well the pupils performed. The analysis tool allowed us to view the answers given for each task. We then grouped them into categories manually (Figure 3, left), for example revealing that the majority of players solved task 4.6 correctly. About a third made a mistake regarding the decimal digit (7200 or 7.2 instead of 720) and only one of ten pupils was completely wrong. This information then could be used by the teacher to discuss common mistakes in a subsequent lecture. In order to get a hint for the design of similar games the authors also check whether the pupils used a help function summarizing the learning content and giving the pupils hints. This was the case for almost every pupil; a majority even used the function more than once (Figure 3, right).

6. Conclusion

The use case showed us that making internal game information visible to the authors in an easily understandable way is crucial for evaluating the intended effects (e.g. learning or training) and usability of serious games. If the data collection is tailored towards a specific game or technology, it is possible to automatically extract semantic information and reduce the overhead of the study at the same time. Offering aggregated data is great to get a quick overview, while providing a more detailed view for further analysis is crucial in the case of anomalies. When these different views are based on the same data, it is beneficial if they are provided by a suite of closely linked tools or even a single application.

Using a model shared between the games created with a specific authoring tool, our evaluation environment is providing a middle ground between general purpose and highly specialized evaluation tools, balancing information detail and associated costs. While it is tailored to games created with the authoring tool, the general approach could be adapted for other models and environments as well.

Future work will include the analysis of games that use the adaptive features more extensively, allowing us to evaluate the adaptation engine and algorithms themselves. We are also planning to offer the possibility to visualize aggregated data directly without using an external spreadsheet application. And since the evaluation framework is using the same models and data structures as our authoring tool, it would be interesting to couple them even further – for example by feeding the average completion time for each task back into the authoring tool. In a case where the players took more time than estimated, the author could then use this data to decide which tasks should be omitted in order to reach the expected completion time in a future version of the game.

Acknowledgements

Parts of this work where funded and supported by the “Forum für interdisziplinäre Forschung” at Technische Universität Darmstadt in the project “Effekte mathematischer Lern- und Diagnoseumgebungen mit spielerischen Elementen” (May 2011 – September 2012).

References

Bartle, R., 1996. Hearts, Clubs, Diamonds, Spades: Players Who Suit MUDS. *Journal of Virtual Environments*, 1(1), p.19. Available at: http://www.mud.co.uk/richard/hcds.htm.

Bruder, R. et al., 2004. Third party certification of computer-based learning environments. In *EISTA: Int. Conf. on Education and Information Systems: Technologies and Applications*. Orlando, Florida.

Göbel, S. et al., 2009. 80Days: Adaptive Digital Storytelling for Digital Educational Games. In Y. Cao et al., eds. *Proceedings of the 2nd International Workshop on Story-Telling and Educational Games (STEG'09)*. CEUR Workshop Proceedings.

Göbel, S. et al., 2010. Personalized, Adaptive Digital Educational Games using Narrative, Game-based Learning Objects. In *Proceedings of Edutainment 2010*. pp. 438–445.

Hilbert, D.M. & Redmiles, D.F., 2000. Extracting usability information from user interface events. *ACM Comput. Surv.*, 32(4), pp.384–421. Available at: http://doi.acm.org/10.1145/371578.371593.

Hullett, K. et al., 2012. Empirical analysis of user data in game software development. *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement - ESEM '12*, p.89. Available at: http://dl.acm.org/citation.cfm?doid=2372251.2372265.

Ketkar, N.S. & Youngblood, G.M., 2010. Graph-Based Data Mining for Player Trace Analysis in MMORPGs. In *Game Programming Gems 8 (AI Section)*. Charles River Media, pp. 335–352.

Kim, J.H. et al., 2008. Tracking real-time user experience (TRUE): a comprehensive instrumentation solution for complex systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, pp. 443–452. Available at: http://doi.acm.org/10.1145/1357054.1357126.

Korossy, K., 1999. Modeling Knowledge as Competence and Performance. In D. Albert & J. Lukas, eds. *Knowledge Spaces: Theories, Empirical Research, and Applications*. Psychology Press, pp. 103–123.

Liu, Y.-E. et al., 2011. Feature-Based Projections for Effective Playtrace Analysis. In *Proceedings of the 6th International Conference on Foundations of Digital Games*. New York, NY, USA: ACM, pp. 69–76. Available at: http://doi.acm.org/10.1145/2159365.2159375.

Medler, B., John, M. & Lane, J., 2011. Data Cracker : Developing a Visual Game Analytic Tool for Analyzing Online Gameplay. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: ACM, pp. 2365–2374. Available at: http://doi.acm.org/10.1145/1978942.1979288.

Mehm, F. et al., 2010. Bat Cave: A Testing and Evaluation Platform for Digital Educational Games. In *Proceedings of the 4th European Conference on Games Based Learning*. pp. 251–260.

Nacke, L.E. et al., 2009. Playability and Player Experience Research. In *Proceedings of DiGRA 2009: Breaking New Ground: Innovation in Games, Play, Practice and Theory*. DiGRA. Available at: http://www.digra.org/dl/display_html?chid=http://www.digra.org/dl/db/09287.44170.pdf.

Nacke, L.E., Drachen, A. & Göbel, S., 2010. Methods for Evaluating Gameplay Experience in a Serious Gaming Context. *International Journal of Computer Science in Sport*, 9, p.2010.