Innovation Methods in Story-Driven Games: Genre Variation

Clara Fernández-Vara, Massachusetts Institute of Technology

Abstract

The paper proposes a method for game design innovation in story-driven games, as exemplified by the development of the adventure game prototype *Rosemary*. This method selects a game model in which a specific variation is introduced. Developing a game where the interface, interaction design, rules, goals, and themes are all new can be overwhelming for the user as well as the developers, so using a pre-existing model can ground the development and help evaluate the success of the innovation introduced.

The design method proposed is called Genre Variation. This methodology relies on a particular story-driven game model as the foundation to introduce new mechanics. After selecting the model, the next step is identifying a design problem that has not been tackled before. Then the variation is implemented as a game prototype and evaluated, following the principles of iterative design. In this case study, the problem was designing the mechanics of memory, and how to turn remembering into a core mechanic of the game. This method is intended at facilitating game development within the limited resources of academia.

Keywords

videogames, game design, story-driven games, methodology, experimentation, innovation

Academic games research can take two basic approaches: critical and development-driven. The critical approach means studying pre-exisiting games, their formal qualities as well as their socio-cultural environment. Games research can also be integral to game development. Rather than expecting games (commercial or not) to bring about innovation, games researchers can work on expanding the design space of digital games. As Mateas and Stern (2005) put it:

"[... M]aking games is required to discover new regions in design space, to understand the relationship between the game architecture and design space, and to probe the local islands that have already been partially explored through previous desings".

The motto proposed by Mateas and Stern (2005), "build it to understand it," is useful to learn more about the dependencies of videogames on the technology (the game architecture), as well as to find the gaps in the design space where

innovation can take place. A hands-on approach does not discount the value of theory and criticism. In fact, a well founded understanding of current game design helps identify possible avenues of innovation as ways of expanding the pre-existing islands within the design space.

In an academic setting, videogame innovation depends on a series of trade-offs. Academia has the freedom to choose what aspects to explore, without the time constraints that commercial developers have to deliver a product. Even when game researchers fail to produce a successful game, they can still learn something from the process and apply it to the next project. While academics do not have a large monetary investment to lose, they usually lack the resources commercial developers have. When the technology is available to universities, researchers cannot have as much time and people working on a single project as a commercial developer. Therefore, game research projects tend to be smaller in scope than most mainstream commercial games. If the scope of a game for research is not adequately defined, the projects are usually unfinished and unreleased, so the results of the research may either be incomplete or only available on paper, so that it is not possible to contrast them with the actual artifact.

An additional issue is developing games with a story component, which can be more resource-consuming than other videogame genres. Story-based games, such as computer role-playing games or point-and-click adventure games usually require a larger amount of assets: graphics, animations, sound and text.[1] The audiovisual aspects are essential to represent the story components, and the story is usually inextricable from gameplay—the plot unfolds as the player advances in the game. This also makes it difficult to use methods of rapid prototyping, since even a paper prototype (e.g. a pen-and-paper roleplaying game) can demand more preparation than an arcade game, for example, for which a few game tokens and a board may be good to start playing a first version of the game (see Fullerton, Swain & Hoffman, 2004, pp. 187-182).

The intersection of story and game in story-driven games has the potential to generate new narrative forms and novel gameplay experiences. Much has been written about the limitations of story-driven games and how they restrict the player for the sake of having the player complete a linear sequence of events (Aarseth, 2004). However, there is also the potential to generate new games that defy those limitations. Building experimental games can expand the design space of videogames, and story-driven games in particular.

This paper proposes a methodology to foster innovation in story-based games through development in an academic setting. The method is called Genre Variation, because it incorporates a new set of rules or system within the pre-existing conventions of a specific story-driven genre. This method is evaluated by examining the development process of the adventure game *Rosemary*. This game was made by a team of undergraduate students led by a researcher, and is

available online (http://gambit.mit.edu/loadgame/rosemary.php). The method here expounded covers the process of inception, development and assessment of the game through playtesting.

Method: Genre Variation

The method consists of choosing a model within a story-based genre, and introducing a new set of rules within that game. Modifying a game is a known exercise to design games (Salen & Zimmerman, 2004, pp.18-19; Brathwaite & Schreiber, pp. 253-4). To turn the exercise into something productive, developers must have a design focus, defined by a series of parameters that delimit the design issues that they will be addressing (Salen & Zimmerman, 2004, p.16). The two main parameters that help define the design focus in the Genre Variation method are the game model and the design problem.

The Genre Variation methodology consists of three steps: defining the model, specifying the variation, and iterative development.

Defining the Model

The game model in which the variation is introduced must be very specific, usually defined by a game or series of games. In the present case study, the model chosen by the researcher was point-and-click adventure games which used the SCUMM engine (Script Creation Utility for Maniac Mansion), such as *Maniac Mansion* (Lucasfilm Games, 1987), *The Secret of Monkey Island* (Lucasfilm Games,1990b) or *Full Throttle* (LucasArts,1995). Other possible ways to define the model can be a role-playing system such as *Dungeons and Dragons 3rd edition* (Cook,Tweet, & Williams, 2000), or choosing a single pre-existing game.

One of the advantages of using a pre-exisiting model is that it is associated to a series of conventions, which players may already be familiar with. It also saves time in the inception of the game, since the model brings a set of design decisions that researchers can build on, such as the user interface, the navigation in the space, or the standard actions to interact with the game.

The SCUMM model uses a point-and-click interface (see Figure 1). The player selects an action from a list of verbs in a menu, and then selects the object or character on the screen in which to perform it, such as "take key" or "open microwave". The player thus constructs a sentence, which is a command for the player character. The player character then carries out the action or responds that it was not possible to do it (e.g. "I don't think that's a good idea").

Figure 1: The SCUMM engine model: *Maniac Mansion* (Lucasfilm Games, 1987)

In the SCUMM model, the player character does not die. That means that the game is not over until the player hs completed all the puzzles; the player can make mistakes without ending the game before experiencing the whole story. Thus, the player is free to explore the world and experiment in it, in order to figure out how it works. In other adventure game models, such as Sierra's AGI or SCI models, the player character could die in certain parts of the game if the player made the wrong choice. For example, in Sierra's *Space Quest: The Sarien Encounter* (Sierra Online, 1986), if the player forgets to fasten the seatbelt before starting a space trip, the player character will die and bring the game to a premature end. Eliminating the death of the player character encourages players to complete the game, rather than abandoning it after dying in the same area several times in a row for no clear reason. The game manual of *Loom* (Lucasfilm Games, 1990a) explains the design philosophy of the SCUMM games:

"We believe that you buy our games to be entertained, not to be whacked over the head every time you make a mistake. [...] We think you'd prefer to solve the game's mysteries by exploring and discovering, not dying a thousand deaths." (Loom game manual, Lucasfilm Games, 1990a, p. 7)

The SCUMM model provides a specific type of story-driven game: point-and-click graphical adventure, with menu-driven commands, which encourages the player to explore and finish the game, rather than being punished whenever the player makes a mistake.

Since SCUMM is a proprietary engine and not available for non-commercial developers, the team chose Wintermute, an engine that incorporates many of the SCUMM features. Wintermute is freely available online (http://dead-code.org/), although it is not open-source.

Game engines are a double-edged sword. They speed up the process of implementation of a game, thanks to ready made code and samples. Engines

also enforce the conventions of a specific genre which are already built into the system: e.g. having a player character, and a point-and-click interface to command the character what to do. This allows making more complex games, since the developers do not have to implement every single feature from scratch. Conversely, the ready-made blocks can also get in the way of innovation, since often they reflect a series of design decisions the developers have no control over. For example, Wintermute did not incorporate a verb menu by default, so the programmers of *Rosemary* had to perform a series of workarounds to implement the menu-driven User Interface.

Specifying the variation

The next step of the method is identifying a problem within the design space, which can be addressed within the possibilities of the chosen model. The problem will be addressed creatively by introducing a variation in that model, adding or modifying a set of rules that will alter specific aspects of the game. These new rules can be a new set of mechanics, i.e. rules that refer to the way the player interacts with the game (Sicart, 2008), or a new way to generate behaviours within the game (e.g. reactions of non-player characters that depend on whether the player has been kind to them in previous interactions).

The methods to specify the variation can be multiple—from a systematic survey of games, to an ethnographic study of players of a specific game or genre. Different research approaches will help identify the shortcomings of the model.

For *Rosemary*, the variation arose from an game idea the researcher had, in which memories resurfaced by revisiting a place one has not been to in a long time. Researching through a variety of adventure games indicated that although memory had been frequently used as a theme in adventure games (Thomas Disch's *Amnesia*, Cognetics Corporation, 1986; *Trace Memory*, CING 2005), there were no specific mechanics that modelled how memory works. This became the research issue of the project, creating a game where "remember" was the core mechanic of the game, and a means to obtain information from the fictional world of the game. "A core mechanic is the essential play activity players perform again and again in a game." (Salen & Zimmerman, p. 316). The adventure games that have dealt with memory as a theme do not have "remember" as one of the main activities in the game. Remembering is usually reduced to providing the player with information that she was not aware of through a description or a cut-scene, but there are no specific interactions that model how humans remember information.

The novel system had to simulate the player character's memory, not the player's. Remembering as a core mechanic had to consist of helping the character recall information, things that the character already knew, but she had not thought of in a long time. The researcher established a further restriction: amnesia could not be used as a device, either in the story or as part of the

mechanics. Amnesia is a trite excuse to equate the knowledge of the player with the character's, and has been used too often in videogames.

By defining the game model and the problem the variation will address, researchers can scope of the game by identifying the design elements that will need the most work. The goal is to produce a finished playable prototype at the end of a specific period of time, in order to be able to test and later demonstrate the innovation. Choosing a game model provides a set of design decisions that the developers do not have to make. However, these ready-made design decisions can also be an obstacle, since part of the innovation will likely consist of reforming or even tearing down some of these conventions. For example, in *Rosemary* it would not have been enough to add an action "remember" to the menu, because that alone did not reproduce a system to help the player remember; "remember" as an action required more nuance.

Iterative Design: Bringing the player into the game

The success of the innovation introduced cannot be determined without evaluating player's experience. A game is not whole until the player experiences it; the player's interaction completes the game system. Thus, the figure of the player must be taken constantly into account during development. Iteration and playtesting are inextricable from the Genre Variation method (and from game development in general).

Iterative design is a methodology that allows developers to include the player in the game design process, and it is one of the most popular approaches to current game design (see Salen & Zimmerman, 2004; Fullerton, Swain & Hoffman, 2004; Braithwaite & Schreiber, 2009). Salen and Zimmerman describe this practice:

"Emphasizing playtesting and prototyping, iterative design is a method in which design decisions are made based on the experience of playing a game while it is in development. In an iterative methodology, a rough version of the game is rapidly prototyped as early in the design process as possible. This prototype has none of the aesthetic trappings of the final game, but begins to define its fundamental rules and core mechanics. [...] This prototype is played, evaluated, adjusted, and played again, allowing the designer or design team to base decisions on the successive iterations or versions of the game. Iterative design is a cyclic process that alternates between prototyping, playtesting, evaluation and refinement." (Salen and Zimmerman, 2004, p.11).

Iterative design invites a constant evaluation of the game, helping both developers and researchers to reflect on the design process. It encourages experimentation (change the game, evaluate in playtesting, change again), which makes it a particularly useful methodology to build games as research. Every

iteration is a small experiment, whose results are incorporated into developing the game further.

Rosemary's iterations were tested with players who were not familiar with the game. There were three formal playtesting sessions, spread out through the 7 months of development. In the first session, most of our 11 players were experienced and familiar with the adventure game genre; the second set of playtesters were a group of 20 teenagers, who did not seem familiar with the genre. These two groups tested the implementation of the novel mechanics. The last group of 13 playtesters helped us ratify our design decisions before finalizing the game. All players were observed by the researcher and the development team while playing, and responded a written questionnaire about their experience after they finished. The evaluation of playtesting was qualitative (evaluating how players understood the game) instead of quantitative (e.g. asking players to qualify different aspects of their experience in a scale 1 to 5).

The problem of story-based games is that they can be difficult to prototype rapidly.[2] Since gameplay is dependent on the story, the story must be at least outlined before starting to play. Putting together a rough paper prototype, with a board and a few tokens, is a perfectly valid method for many other genres (see Fullerton, Swain and Hoffman's (2004) examples of how to prototype a first person shooter). The preparation time needed for a rough prototype in a story-based game can take much longer, since it needs building a fictional world, how it works, and how to interact with it. The "aesthetic trappings" that Salen and Zimmerman mention are necessary to construct that world, even if they are assets borrowed from other games or early versions of the actual asset. The first prototype of *Rosemary* was ready after six weeks of development, which can be a long time for an academic project. Thankfully, the prototype was relatively successful; if it had not, the team would have lost much of the work done until that point.

The team who made *Rosemary* managed to incorporate iterative design by generating builds that added, eliminated or revised features. Temporary art and incremental expansions in the game helped advancing the production and evaluating it periodically. The following sections explain the iterative process of the game, focusing on the novel aspects of it: the memory mechanics that modelled how the player character remembers people and events.

***Rosemary*: The Development Process**

The first step of development was determining how memory worked in the game, which is part of defining the variation on the mechanics. As part of the preparation for development, the researcher presented the team with different models of how memory works, such as Pinker's comparison of the human mind with a computer (2005). The developers also went through several brainstorming sessions, in which they reviewed different events and actions related to human

memory that may serve as the premise of the game. For example, the team discussed the Rashomon effect, in which the same events are remembered differently by a series of witnesses; they also considered how Alzheimer affects memory, or how humans remember in fragments rather than holistically.

Mnemotechnics, i.e. techniques and principles to organize memories and improve recall, were the key to introduce novelty in the game. The techniques are already a system, which makes them easier to formulate as game rules. Of the catalogue of mnemotechnics, Quintillian's concept of the Memory Palaces appeared to be the most useful as a set of potential mechanics, as well as the most evocative to make a game (Quintillian, 1920, Book XI, Chapter 2). Quintillian's method was meant to help an orator to commit his speeches to memory by turning each topic of his speech into a symbol, which would be then placed in an imagined house or other space he was familiar with. The orator would then traverse the space mentally; by recalling where each symbol was, he reproduced the speech that he had prepared.

Quintillian's Memory Palaces provide a productive model because it is already formulated as a set of mechanics, i.e. arranging memories and associating them with specific locations in order to recall the information. The association of memory with moving through a space also made it a particularly convenient model for videogames, because its implementation can take advantage of the spatial properties of digital environments, as described by Murray (2001), which allow the player to navigate a virtual space. The Memory Palaces could also benefit from the SCUMM model, which encourages players to explore the the game and try arranging memories in different ways without getting punished.

Story Premise

The next step was coming up with a story premise that would suit the concept of the Memory Palaces. After a couple of brainstorming sessions, the team found an intriguing premise that would set in motion the events of the game. The player character finds a photo that evidences that her imaginary childhood friend was a real person. She comes back to her home town, which she has not visited in many years. Nobody lives there any more, so the player explores the environment to trigger off her memories. The goal of the game would be to find out what happened to her friend by remembering people and events that may yield the right clues.

Modelling Memories

The game design evolved from the concept of the Memory Palace and the story premise, so that story and design advanced in conjunction. The mechanics of remembering were implemented as two different features: comparing between the character's memories and the present, and the Memory Palace itself.

Comparing Past and Present

The player can compare what the town looks like in the time the character is in with the way that she remembers it. A switch allows the player to go back and forth between past and present; comparing both versions allows the player to find differences that lead to new information. This feature was relatively easy to implement programatically, although it also doubled the amount of time needed to produce the visual assets. Every background and object needed two different versions, one for the memory and one for the present. The team was very confident they could produce the assets in the alloted time; and the mechanic was novel enough in adventure games that the feature was greenlit. Unfortunately, the estimates of the artists in the team were too optimistic; the assets were produced for four different locations, but the amount of time it took to produce the background of each location made it impossible to expand the number of locations in the game.

The memory/present comparison combined well with the Memory Palace. The player character would remember things by interacting with the environment. When a specific object or person triggered a memory, a symbol representing it would appear in the Memory Palace, which the player could manipulate there. Conversely, as the player solved puzzles in the Memory Palace, new objects and characters appeared in the memories of the town, allowing new interactions with the environment. For example, solving one of the puzzles in the Memory Palace made a lunchbox appear in the memories, which allowed the player to find the lunchbox in the present. It fit well with Quintillian's observation about memory and spaces (1920): "For when we return to a place after considerable absence, we not merely recognise the place itself, but remember things that we did there, and recall the persons whom we met and even the unuttered thoughts which passed through our minds when we were there before." Comparing memory and present aimed at reproducing how remembering events, people, or things may help us see the present in a new light and understand current events better.

During playtesting, this feature proved to be rather successful from the beginning. In every testing session, most players understood that they were comparing two versions of the same location, and that they were past and present. In the second evaluation, when all the final backgrounds were in place, only one player out of twenty thought that the difference was day and night (see Figure 2). Otherwise it the mechanic was easy to understand and play with.

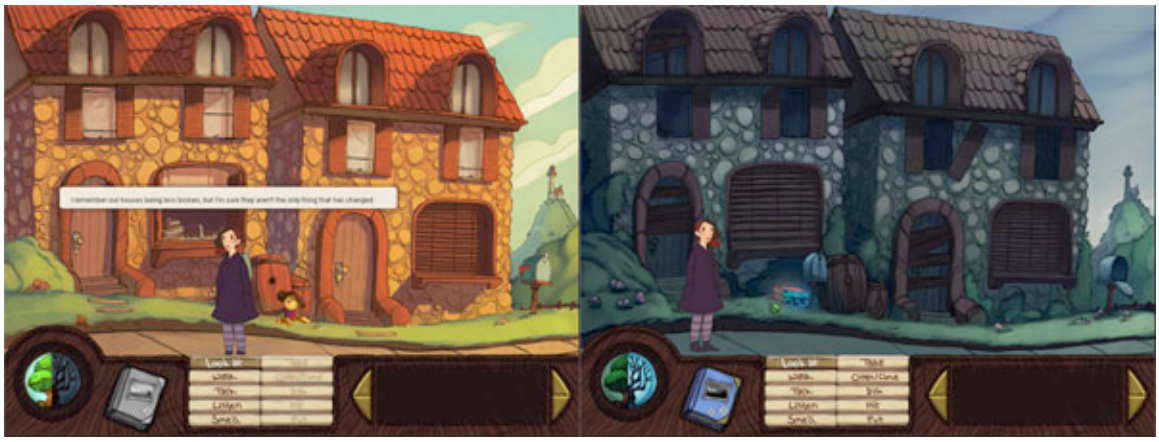

Figure 2: Comparison between memories and present in the final version of the game.

The Memory Palace

The Memory Palace represents the memories of the player character, where they are stored and arranged in order to allow the character to remember information. The mechanics of the Memory Palace primarily consist of arranging and connecting representations of memories.

The first version of the Memory Palace was presented as the interior of a tree house, where the character used to play with her friend (see Figure 3). This was a metaphor, not an actual location in the fictional world of the game. The tree house had a workbench at the front, where the memories would be placed in juxtaposition. If the memories were not connected, nothing happened; if they were, they would be cleared and a new memory would appear.

Figure 3: First version of the Memory Palace: the tree house.

This implementation was problematic, as the first playtesting session evidenced. Seven out of eleven players believed the Memory Palace was a real location in the game, not a metaphorical space—the outside of the treehouse was one of the locations of the game. The player could access the Memory Palace by clicking on one of the buttons in the menu at any time. The problem was that the button only appeared after visiting the outside of the tree house, in an attempt to introduce players to the game one feature at a time. Only four players understood the tree house as the representation of the memories of the character, the rest were not able to establish that relationship.

Another problem was how memories were connected. The player had to place the memories on the workbench; if they were related, they would disappear and make another memory appear in the tree house. In order to show the relationship between two objects, the player had to place them next to each other. Players were rather confused, because the puzzle that taught the the player the mechanics of the memory palace was connecting a trunk and a key. The point-and-click interface seemed to players to drag the key on the chest, trying to use them together, but that was not the right solution.

In short, playtesting evinced that the problems of the first version of the Memory Palace were its unclear metaphorical status with respect to the rest of the game, as well as a confusing interface, in the form of the workbench. Players figured out most puzzles through trial and error rather than understanding how the game worked, which was not the experience intended by the developers.

The re-desing of these mechanics kept the tree house metaphor (see Figure 4), but revised the mechanics of the Memory Palace. The player had to place the memories from the bottom of the screen on the right area of the house, such as a table or a shelf. When two related items were next to each other, they would stay in place, and make a new memory appear. The trunk and key pair was eliminated, since it did not make sense to put a key next to the chest to connect them. Arranging memories in space was closer to Quintillian's description, where the memory had to be in the right place in order to be able to remember. On the other hand, their implementation was somewhat problematic. While the mechanics were easy to implement, the visual assets were very specific. The place of every puzzle had to be defined beforehand; if any puzzle had to be changed or moved, all the visual assets had to be re-done. This would have hampered any further iterations and expansions on the puzzles.

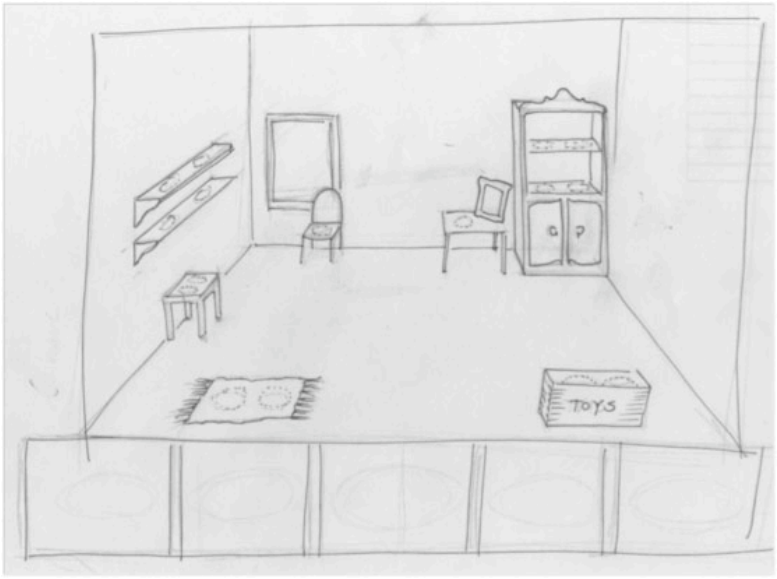

Figure 4: Sketch of the revised Memory Palace as a tree house.

The solution was keeping the mechanics (place the memory in the right area) and changing the metaphor. Rather than using an architectural space, the memory of the player character is represented by a photo album, where a photo stands for each memory (see Figure 5). Every page of the album has two empty slots to put the photos in. Each photography slot has a caption where four words were missing; each photo has two words associated with it, which would fill two of the empty slots. If two related photos are in the right slots, the puzzle is solved, the caption is be completed correctly, and a paragraph explains what the character remembers, providing more information to the player.

The photo album was a success during playtesting, since placing photos on their right slot was intuitive enough to most players; the Mad Libs-like type of mechanics provides further feedback to the player by indicating whether the photos are on the right slot. It was also a more appropriate metaphor—after all, photos are used as mementos of times past. In the final playtest, the photo album was interpreted in a variety of ways, which still helped players complete the game. Five out of thirteen players thought it was a journal where the character noted what happened to her; one interpreted it as "the puzzle of her life". Even if it did not use Quintillian's metaphor, it was also more true to the description of how it worked. After all the slots had been filled, the player can read a series of short stories that tell more of the past events in the town.

Figure 5: Final version of the Memory Palace as a photo album.

Evaluation of the Genre Variation Method

The Genre Variation method demonstrates that player experience is inextricable from the game design process, even when the initial focus of innovation is on the mechanics of the game. The game mechanics are inherent to generating player's experience, so that the degree of innovation and effective design cannot be assessed without playtesting.

Rosemary was a first experiment to evaluate whether it was possible to produce a brief adventure game in an academic environment. The innovation seems successful, since players quickly grasped the mechanics of memory by the second playtest. More significantly, younger players (15-17 years old) seemed to understand the novel mechanics better than the conventional mechanics of adventure games. These players thought that they could control the player character as if it were *The Sims* (Maxis Software, 2000), where the player clicks on a button and the character performs the action, rather than constructing a command by clicking on the menu and then the object on which to operate.

Genre Variation was a successful method to produce a story-driven game in a relatively short period of time (7 months during the academic term, 10 hours a week) with a small team (5 to 7 people at a time per semester). *Rosemary* can be completed in 15 to 20 minutes depending on the familiarity of the player with the adventure game genre. Providing a game that is polished and complete, albeit brief, also produces a much more welcoming attitude during playtesting, particularly in younger players.

The choice of game engine facilitated production, although it also created a series of bugs over which the development team had no control over. The most notable were some rare rendering problems, where the graphics would not load

properly or display the wrong backgrounds. This affected the final polish of the game, although did not cripple playtesting.

The problem with this methodology is that it only evaluates one specific instance of the game, rather than comparing a set of different novel solutions to the same problem. This requires making several versions of the same game, or different games about the same problem, which is difficult with the limited resources of an academic laboratory. However, the experience of *Rosemary* has proved that early prototypes can also yield useful playtesting data. The new project based on Genre Variation the researcher is working on is based on building smaller prototypes, using different mechanics based on the same problem, which is modelling the logic of dreams. Each prototype takes the lessons learned on the previous one, and expands on the mechanics. The iteration happens in each game and from game to game, and they are all addressing the same problem.

The Genre Variation method aims at spurring creativity by helping define the problem that must be solved. Game design is thus similar to solving puzzles, a meta-game that developers can also play inventively.

References

Aarseth, E. J. (2004). Genre Trouble: Narrativism and the Art of Simulation. In N. Wardrip-Fruin & P. Harrigan (Eds.), *First Person: New Media as Story, Performance, and Game* (pp. xiii, 331 p.). Cambridge, Mass.: MIT Press.

Brathwaite, B., & Schreiber, I. (2009). *Challenges for Game Designers: Non-Digital Exercises for Video Game Designers*. Boston: Charles River Media.

CING. (2005). *Trace Memory*: Nintendo of America.

Cook, M., Tweet, J., & Williams, S. (2000). *Dungeons and Dragons 3.0: Player's Handbook: Core Rulebook I*: Wizards of the Coast.

Cognetics_Corporation. (1986). *Amnesia*: Electronic Arts.

Fullerton, T., Swain, C., & Hoffman, S. (2004). *Game Design Workshop: Designing, Prototyping and Playtesting Games*. Berkeley, CA: CMP Books.

Lucasfilm Games. (1987). *Maniac Mansion*: Lucasfilm Games.

Lucasfilm Games. (1990a). *Loom*: Lucasfilm Games.

Lucasfilm Games. (1990b). *The Secret of Monkey Island*: Lucasfilm Games.

Mateas, M., & Stern, A. (2005). Build It to Understand It: Ludology Meets Narratology in

Game Design Space, *DiGRA*. Vancouver.

Maxis Software. (2000). *The Sims*: Electronic Arts.

Murray, J. H. (2001). *Hamlet on the Holodeck: The Future of Narrative in Cyberspace*. Cambridge, MA: The MIT Press.

Pinker, S. (2005). How Does the Mind Work. *Mind and Language, 20*(1), 1-24.

Quintillian. (1920). Institutio Oratoria, *Loeb Classical Library* (Vol. IV). http://penelope.uchicago.edu/Thayer/E/Roman/Texts/Quintilian/Institutio_Oratoria/11B*.html#2

Salen, K., & Zimmerman, E. (2004). *Rules of play: Game Design Fundamentals*. Cambridge, Mass.: MIT Press.

Sicart, M. (2008). Defining Game Mechanics, *Game Studies* (Vol. 8).

Sierra On-Line. (1986). *Space Quest: The Sarien Encounter*: Sierra On-Line.

Footnotes

[1] An exception to the rule is interactive fiction (IF) / text adventure games, where all the assets are textual.

[2] Again, Interactive Fiction may be an exception.

Clara Fernández-Vara

Clara Fernández-Vara is a Postdoctoral Researcher at the Singapore-MIT GAMBIT Game Lab. Her work concentrates on adventure games, as well as the integration of stories in simulated environments. She is particularly interested in cross-media artifacts from the standpoint of textual analysis and performance.