Game Design

Individual Reflection on *"Sun Valve"*, made by Team *Scylla and Charybdis*

Simon Cutajar

December 14, 2011

Contents

Chapter 1

Introduction

When confronted with having to design a game for my *Game Design* course, the whole team decided to take a different approach to the style and theme of the game being designed.

SCYLLA & CHARYBDIS

(a) *Scylla and Charybdis* Logo

Sun Valve

(b) *Sun Valve* Logo

Figure 1.1: Group Logo and Game Logo

The game *Sun Valve* is an adventure game with a strong emphasis on narrative that is based around the moral dilemma of the Experience Machine, originally presented by Robert Nozick, as an ethical choice that must be made during the game. During the design and production of the game, games such *January* (Vreeland, 2011) and *p0nd* (Peanut Gallery, 2010) served as our inspiration, but it was surprising to see our game being compared to other games such as *Myst* (Cyan, 1993) by other people.

Chapter 2

Game Design

2.1 The Process of Making a Game
or "Sometimes It's Best To Plan Things Through"

As stated by (Edwards, 2006), there are 4 main phases in the production of games. These are:

- *concept* phase – where the idea behind the game is thought up
- *pre-production* phase – a comprehensive design document, amongst other documents necessary for the game, is written
- *production* phase – where agile methods such as SCRUM are used to get things done
- *post-production* phase – testing occurs and bugs are fixed

It is interesting to see how as a group, we structured our time and our development process around these phases.

2.1.1 Game Concept

The only thing that was pre-decided before brainstorming ideas was the fact that we wanted the game to contain meaning and not purely for entertainment. We used several different brainstorming techniques, such as mind maps (as seen in Fig. 2.1) and individual presentations. As stated in (Fullerton, 2008), we limited the amount of time taken for brainstorming and idea generation, making frequent use of the whiteboard (as can be seen in Fig. 2.2). One particular thing I noted is the fact that during the brainstorming sessions, the team focused on generating as many ideas as possible, regardless of whether they were relevant or not. Also, the team did not judge any ideas until after the brainstorming session, allowing people to continue participating (Michalko, 2006).

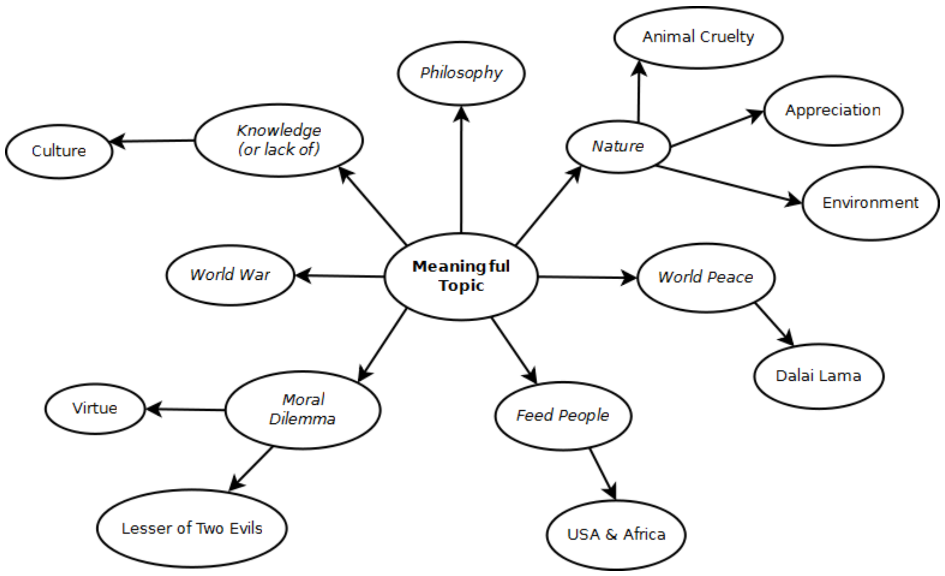

Figure 2.1: Brainstorming Meaningful Topics

It is interesting to note that by unintentionally staggering out brainstorming sessions over the week (in our case, they were held on Monday and Wednesday), we also allowed ourselves to unconsciously continue thinking about the problem (von Oech, 2008).

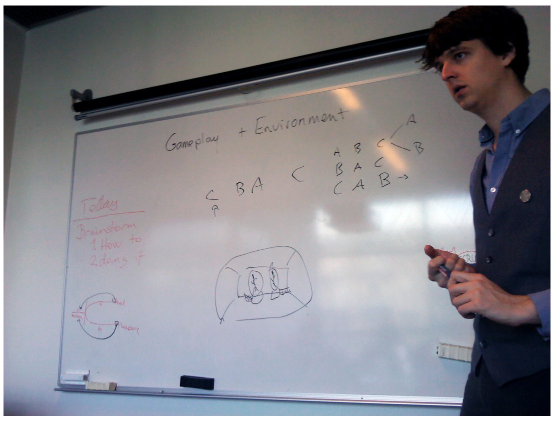

Figure 2.2: Using the Whiteboard for Brainstorming

2.1.2 Entering Pre-Production

It was during the pre-production phase that dedicated team roles were allocated to people. I was given the title of *Lead Programmer*, as well as *Vice Music / Sound Designer*. I was also in charge of posting stuff to the course Tumblr[1]. A storyboard was also drawn up for the game, since the game was intended to be a mainly narrative, exploratory game. We eventually settled on "branching story" structure for the game (as stated in (Brathwaite and Schrieber, 2008)), where the player had to make a choice in the game that effectively split the narrative's direction into two. A theme and setting was chosen for the game's concept, and an art and music style was also decided upon.

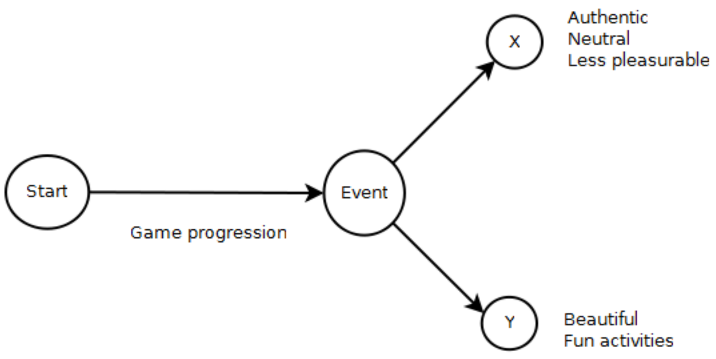

Figure 2.3: Narrative's Flow

Since we decided to use the SCRUM method of producing the game, which involved producing a backlog of all the things that had to be done in order to have a finished product, we also had to decide on the mechanics that we would need in the game. We also had to give an estimate on how long things would take to be completed (Moreira et al., 2010). I was mainly in charge on determine how long tasks related to programming would take.

2.1.3 Production

Production of the game was done using the Unreal Development Kit. Other external programs were used to create assets, such as 3D Studio Max and Maya for 3D assets. As the lead programmer, I was responsible for using UDK to bring the game together, as well as create the mechanics needed for the game. I used several different programming languages, such as UnrealScript, ActionScript, C++ and the Kismet flowchart interface, as well as using the available tools inside UDK to create and modify the game world.

I mainly switched between using Microsoft Visual Studio 2010, Adobe Flash CS5 and the UDK Editor.

2.1.4 Post-Production

After the lead tester in the team successfully came up with a suitable playtest and usability test, it was time to test out our alpha version of the game by making people play it. I was in charge of filming the test subjects while they answered questions and played through the game, as well as logging the actions taken in-game through code. Using the knowledge we gathered during the playtest session, we were able to iron out the more obvious bugs and design a better experience for users.

2.2 Working Together in a Team
or "Other People Have Ideas Too"

As can be seen in Fig. 2.4, each person had their own unique role in the team and therefore had certain responsibilities. This was done so that if things were not going according to plan, or things were not being done, people could be held accountable. However, since the game was a team effort, it was impossible to handle certain things on your own and one had to ask other team members for assistance. As a programmer for example, I had to consult with the lead designer to make sure that the overall vision of the game was being kept.

Working in a team is fundamentally different to working on your own. As stated in (Schell, 2008), the whole team must love the game they are making (as well be able to work professionally with the other team members). As a designer, one must make sure that each team member feels involved. This sometimes involved being objective about my ideas; as much as I thought my idea was brilliant, so did everybody else think the same way about their idea. There had to be some sort of compromise.

It was also important for us to be able to co-ordinate between each other when we were working on different systems within in the same map, since we wanted to avoid merging problems. Unfortunately, this still occurred, particularly because we did not use any type of version control system outside of putting the files in Dropbox.

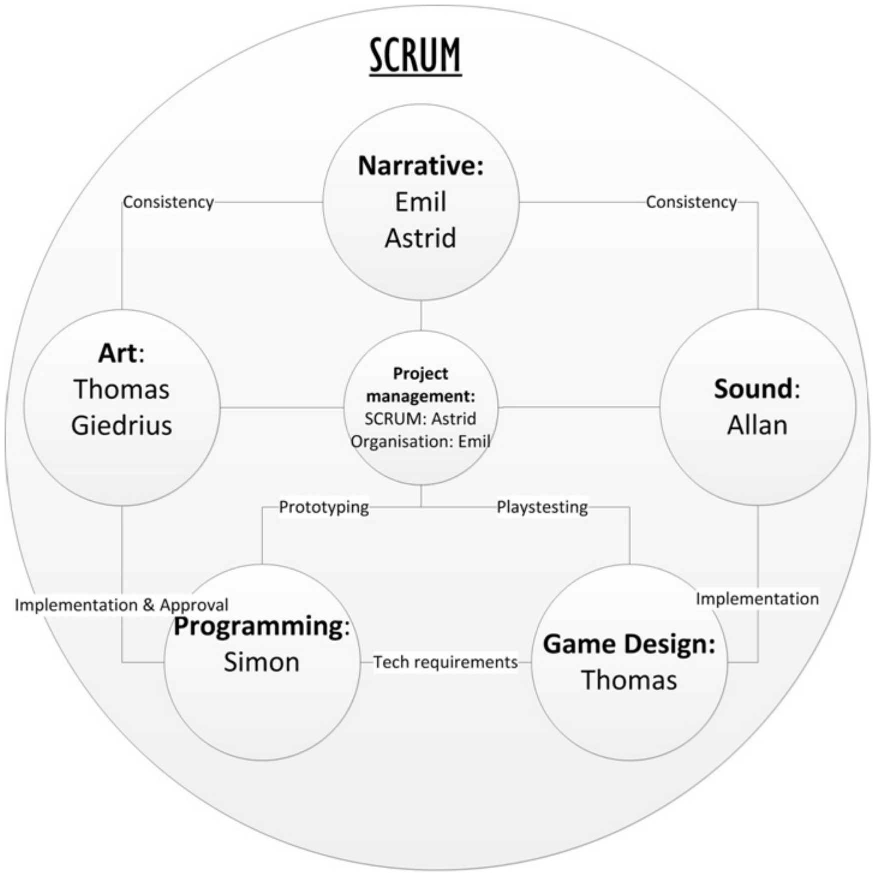

Figure 2.4: Team Structure

2.3 Player Experience and Player Immersion
or "How to Remember That You Design Games for Players"

It was very clear to us to keep the player in mind when designing the different aspects of the game.

It was important to not think of the game as simply a system (and thus continuing the digital fallacy), but also to consider designing the game as an experience. For example, as a programmer, I am trained to solve problems in terms of systems. However, a game does not consist solely of the system, but also of the player's experience, and thus the game must be constructed with the player's experience in mind, as a kind of *second order design* (Stenros and Waern, 2011).

For example, it was intentionally designed for the game not to have a user interface while playing the game. No head-up displays were designed or implemented so as to totally immerse the player in the game environment. We did not need to inform the player about how much life he or she had left, since such information was not used in the game. A minimap was not implemented, since we felt that the island was small enough to not need one. One can say that *Sun Valve*, although not strictly a minimalist game as defined by (Nealen et al., 2011), has minimal design elements by not including a visible head-up display. We also feel that the game has a *low percieved complexity* and a *high systemic complexity*, in that while the gameplay is fairly simple for players to understand, the system behind the gameplay is rather complex.

We made sure that all mechanics were easily accessible by players by adding things such as tutorial splash screen on the player's first encounter with the mechanic. For the gesture system (used in the poetry mini-game and the drawing mini-game), I made sure as a programmer that gestures could be drawn from any starting point and could still be recognized as the gestures we needed. This was done so that players would not break the immersion by having to follow a certain convention for drawing the gesture (such as starting at the top of the screen).

I also made sure that certain affordances were not given to the player when they weren't intended to be given (Stenros and Waern, 2011). Displaying the mouse cursor during the game would automatically make the player think that *Sun Valve* was a point-and-click adventure game. Hiding the mouse cursor and displaying it only during mini-game sequences should inform the player when and when not to use the mouse.

It is surprising to see that even though you feel that the game should be clear enough for users to understand, and that you have taken precautions just in case players decide to do things differently, we were still able to find players who had no idea what they were doing. During testing, we had test subjects who asked us how to control character movement, how to use objects, how to interact with the mechanics and whether the character could run. Although we used a prototype to conduct testing and not a final version, it is interesting to see the results of one playtest. By conducting further tests, we would be able to improve the experience for players even further while still keeping in line to the general idea behind the game.

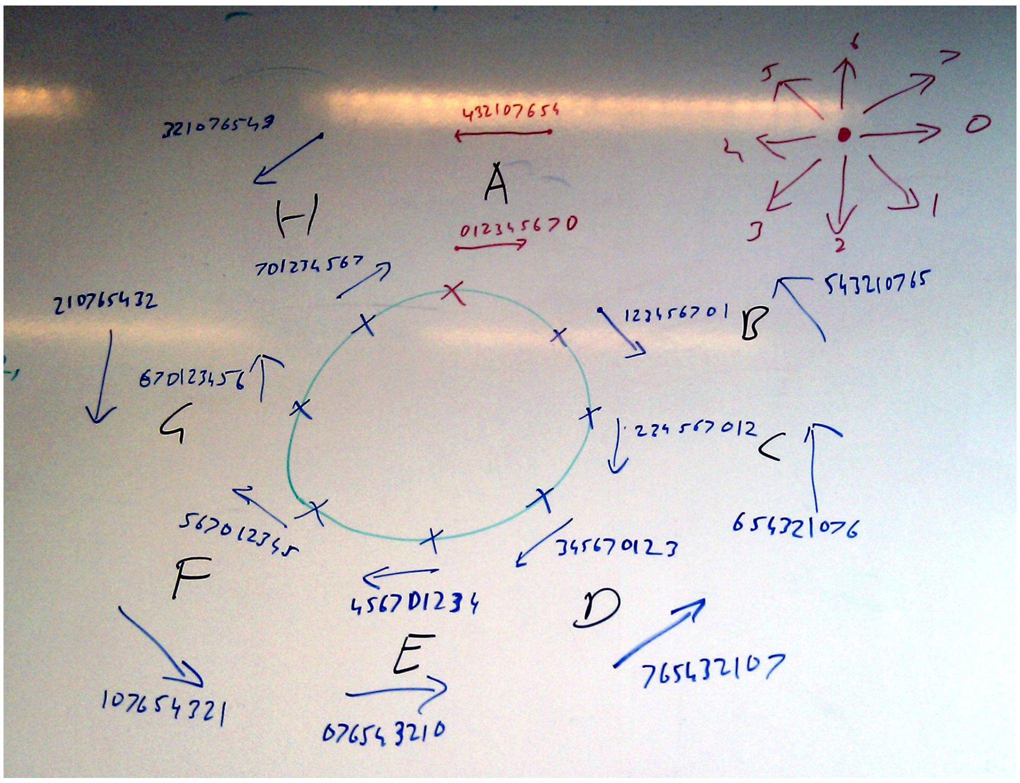

Figure 2.5: Gesture Recognition Design Sketch

2.4 My Experience Working On the Game or "Looks Like it's Going to Be a Long Night of Coding"

I kept a development diary of sorts during the time I spent coding for *Sun Valve*[2]. It's the first time I've ever kept a development diary of sorts, and after I've seen how useful it is, I will definitely be doing it again in the future. Not only is it useful for gathering your thoughts and knowing which decisions you took and why, but it can also be used as a form of public relations if done frequently enough.

My main challenge was the fact that I was not familiar with the Unreal Development Kit. Although I felt that most of the mechanics that we had decided upon were conceptually easy to implement in a programming language, the fact that UDK handled things differently made my progress a lot slower, since I was constantly having to research how things could be done. I felt that reading other people's source code was more useful than following some of the UDK tutorials. I also felt that UDK had a steep learning curve, since as a programmer, one had to know the structure of the already existing

UDK classes before being able to inherit from them and implement my own.

Not being familiar with UDK also meant that it was hard for me to estimate the time that a task would take to complete in order to determine the backlog of tasks for scrum. Programming is inherently tricky to estimate if one is not familiar with the system being implemented or with the tools needed to implement the system, since one could potentially get stuck on a minor problem that takes time to understand and solve. This happened fairly regularly in the beginning of the project, when I was relatively new to UDK, and although I felt it got better towards the end of the project, I feel that I still have a lot to learn about UDK in order to be able to estimate task completion time efficiently. The fact that I was also slightly disappointed with the debugger in UnrealScript (since it did not stop at the breakpoints I set it, and sometimes stopped at lines that did not have breakpoints, making it impossible to debug effectively) only meant that my programming times took longer than necessary.

As the only programmer in the team, I felt I was at a slight disadvantage since I could not discuss my solutions with other programmers in the team. Although the lead designer was experienced in designing and developing games, he did not have the background in programming and algorithm development necessary to be able to discuss the implementation of several features in the game. I feel that having at least one other programmer is beneficial since it allows *pair programming*, a form of programming found in agile development such as *extreme programming*. Pair programming allows for the discussion of implementation ideas, catching bugs as well as keeping focus of the task at hand (Shore and Warden, 2007).

As can be seen in Fig. 2.5 and Fig. 2.6, the team and I made ample use of sketching when trying to figure out how to solve problems and implement mechanics in the game. As stated in (Buxton, 2007), sketching is used as a quick and efficient method to display ideas to other people, and as I got closer to understanding the idea that the lead designer wanted to implement, the sketches became more and more refined. By quickly implementing these sketches in Visual Studio and displaying them as prototypes, I quickly understood the meaning of *prototypes as the manifestation of design ideas*(Lim et al., 2008). It is also interesting to note that the prototyping progression I chose was that of *evolutionary prototyping*, or *iterative prototyping*, rather than *throw-away prototyping*. Therefore, instead of simply building prototypes to understand how mechanics would work, and then discarding them, I adopted an iterative approach where I started with the main product and improved it constantly by iterating over it and adding new features (McConnell, 1996).

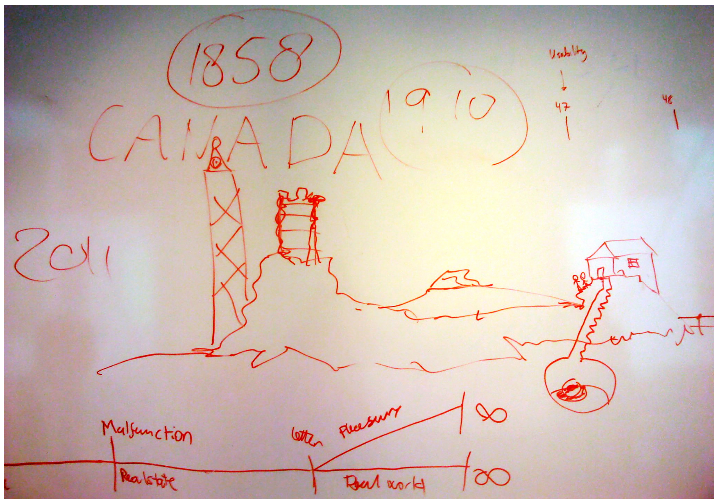

Figure 2.6: A Sketch of the Island

I was also in charge of periodically posting updates on Tumblr, so in some ways, I was in charge of PR for the group. Since the game was heavily based on a narrative, we didn't want to give too much of the story away, so some screenshots were intentionally held back. Apart from putting photos on Tumblr, I also uploaded them to an album on Facebook, as well as linked to them from some tweets in Twitter. This allowed a wide variety of people to hear about the project as well as look at the results we already had. However, we did not focus too much on building a website.

However, as seen in Fig. 2.7, we also decided to attend a Demo Night event[3] held at PROSA; not only to network with other people working on indie games in the Danish scene, but also to show off our game as a work-in-progress. This also allowed us to practice pitching our game, especially to people we didn't know. (Fullerton, 2008) stresses the importance of attending social events such as the PROSA Demo Night in order to network, but also mentions joining appropriate organizations (such as the International Game Developers Association) or attending conferences. I have joined the Danish chapter of IGDA and have attended several Special Interest Group meetings that have been held, so I feel that this allows me to network appropriately. (Nelms, 2000) likens the act of pitching to a performance, always keeping the illusion. I feel that my personal pitching style is closer to a flamboyant, theatrical style than to an overtly serious pitch, but of course, which one is used depends on the product one is pitching.

Figure 2.7: Presenting *Sun Valve* at the PROSA Demo Night

Chapter 3

Conclusions

3.1 Evaluating the Game

As stated in the Introduction, the intention was to develop an exploratory adventure game with a focus on a moral dilemma, a choice that the player must think which will keep him or her thinking. Although the concept of moral dilemmas is present in several other games in passing, such as *Black and White* (Lionhead Studios, 2001), we decided to go for a dilemma that we felt had not been previously explored by video games before. Furthermore, we did not want to develop a video game that contained just the moral dilemma, such as *The Trolley Problem* (Barr, 2011).

It is important to note that although we had around seven weeks to design and develop the game, and we were expected to fail, failure is subjective (Gaver et al., 2009). For example, we succeeded in working together as a team, while I succeeded in learning a new system and developing a game using that system. Whether or not we succeeded in creating a memorable and enjoyable game can be determined using the symptoms for success of failure mentioned in (Gaver et al., 2009). During the playtest for example, 28% made reference to the game *Myst* (Cyan, 1993), while most people commented that they enjoyed their brief test session and would like more time exploring the island. Most people also mentioned that they especially liked the graphics in the game, making reference to the high quality textures used and the scenery (such as trees and the sea). Some test subjects even suggested improvements to the game mechanics based on their experiences.

As to the originality of the game itself, as stated before, there have been many different games that involve moral dilemmas (such as in *Black & White*

(Lionhead Studios, 2001) where the player faces different situations that affect his character in the game, or in games such as *Manhunt* (Rockstar Games, 2003) where the amount of violence that the player uses has no effect in the rest of the game, but is up to the player's judgement)), and many open world and narrative-heavy games (such as *Myst* (Cyan, 1993) or *World of Warcraft* (Blizzard Entertainment, 2004)). However, we place a special focus on character exploration in our game, since when you explore the island in *Sun Valve*, you also explore Conrad, the main character, as person. The player should be able to connect with him as a character. Furthermore, we feel that the type of dilemma being presented to the player is not one that is commonly seen in games.

Notes

[1]Located at `http://gd2011.tumblr.com/`.

[2]Available at `http://scychar.scutajar.com/index.php?title=Simon%27s_Development_Diary`

[3]The Facebook event can be found at `https://www.facebook.com/events/289286391095142/`

Bibliography

Barr, P. (2011). *The Trolley Problem.* [Online Game]. `http://www.pippinbarr.com/games/trolleyproblem/TrolleyProblem.html`.

Blizzard Entertainment (2004). *World of Warcraft.* [PC / Mac].

Brathwaite, B. and Schrieber, I. (2008). *Challenges for Game Designers.* Charles River Media, 1 edition.

Buxton, B. (2007). *Sketching User Experiences. Getting the Design Right and the Right Design.* Morgan Kaufmann, 1 edition.

Cyan (1993). *Myst.* [PC / PS].

Edwards, R. (2006). The Game Production Pipeline: Concept to Completion. IGN Website. `http://games.ign.com/articles/696/696273p1.html`.

Fullerton, T. (2008). *Game Design Workshop: A Playcentric Approach to Creating Innovative Games.* Morgan Kaufmann, 2nd edition.

Gaver, W., Bowers, J., Kerridge, T., Boucher, A., and Jarvis, N. (2009). Anatomy of a failure. how we knew when our design went wrong, and what we learned from it. In *CHI '09 Proceedings of the 27th international conference on Human factors in computing systems.* ACM New York.

Lim, Y., Stolterman, E., and Tenenberg, J. (2008). The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. *ACM Transactions on Computer-Human Interaction*, 15(2).

Lionhead Studios (2001). *Black & White.* [PC / Mac].

McConnell, S. (1996). *Rapid Development: Taming Wild Software Schedules.* Microsoft Press, 1 edition.

Michalko, M. (2006). *Thinkertoys: A Handbook of Creative-Thinking Techniques.* Ten Speed Press, 2nd edition.

Moreira, M. E., Lester, M., and Holzner, S. (2010). *Agile For Dummies.* Wiley Publishing, Inc., CA Technologies edition.

Nealen, A., Saltsman, A., and Boxerman, E. (2011). Towards minimalist game design. In *Foundation of Digital Games Conference Proceedings 2011.*

Nelms, H. (2000). *Magic and Showmanship: A Handbook for Conjurers.* Dover Publications.

Peanut Gallery (2010). *p0nd.* [Online Game]. `http://www.kongregate.com/games/peanutDre/pond`.

Rockstar Games (2003). *Manhunt.* [PS2 / Xbox / PC].

Schell, J. (2008). *The Art of Game Design: A book of lenses.* Morgan Kaufmann, 1 edition.

Shore, J. and Warden, S. (2007). *The Art of Agile Development.* O'Reilly Media, 1 edition.

Stenros, J. and Waern, A. (2011). Games as activity: Correcting the digital fallacy. In *Videogame Studies: Concepts, Cultures and Communication*, page 11. Inter-Disciplinary Press.

von Oech, R. (2008). *A Whack on the Side of the Head: How You Can Be More Creative.* Business Plus, 25 Anv Rev edition.

Vreeland, R. (2011). *January.* [Online Game]. `http://www.colorcave.com/january.html`.